User Guide

Documentation Version 1.6m29

GestureTek®

Table of Contents

Introduction 1

Copyright and Legal Matters 2

Starting Dazzler 2

Dazzler Setup 3

Command Line Options 5

Multiple Displays 6

Setting Up Multiple Displays 6

Blending 7

To configure settings for multiple displays 8

To select a blending option 11

To adjust Universal Blending Interface settings 14

To adjust Color Balance Interface settings 15

Dazzler Logs 16

Exiting Dazzler 16

About Play Items 17

Play Item Attributes 18

Setting Color Values 19

The Duration Attribute 20

Using the Duration attribute 20

Running Play Items 22

Spark Files 22

Nested Play Items 25

Understanding File Paths 26

Using Sample Code 27

Dazzler Demos 29

Hot Keys 29

Play Items - Alphabetically 30

Play Items - Controllers 31

AttractLoopScheduler 32

Attributes 32

Sample: Simple AttractLoopScheduler 32

ChildPlayer 33

Attributes 33

Samples 33

Exit 33

Attributes 33
Sample: Exit 33
Include 34
Attributes 34
Sample: Include within SequentialScheduler 34
Sample: Simple Include 35
RandomScheduler 36
Attributes 36
Sample: Play Random Images 37
SelectionScheduler 38
Attributes 38
Sample: Play Second Child Play Item 38
SequentialScheduler 39
Attributes 39
Sample:IncludewithinSequentialScheduler 39
Sample: Simple SequentialScheduler 40
SparkFile 41
Attributes 41
Sample: Calling Spark File from a Spark File 41
WaitForMotion 42
Attributes 42
Sample: Simple WaitForMotion 43
Sample: Play Spark File on Timeout 43
Play Items - Effects 44
AlphaMask 45
Attributes 45
Sample: Simple AlphaMask 46
AlphaOverlay 47
Attributes 47
Sample: AlphaOverlay 48
AlphaTransition 49
Attributes 49
Sample: Simple AlphaTransition 50
BasicTiles 51
Attributes 51
Sample: BasicTiles Checkerboard 55
Sample: BasicTiles Entire Image Flip 56
Sample: BasicTiles Twirling Strips 57
Sample: BasicTiles Venetian Blinds 58
ColourKeying 59
Attributes 59
Sample: Keyed WaterFX Over an Image 60

Sample: Keyed Flash File Over Thick Gel 61
FishSwarm 62
Attributes 62
Sample: FishSwarm 64
FluidDynamics 65
Attributes 65
Sample: FluidDynamics 66
GenericSwarm 67
Attributes 67
Sample: GenericSwarm 72
Sample: GenericSwarm with Silhouette 73
Greyscale 74
Attributes 74
Sample: Greyscale Shader Filter 75
IceCrack 76
Attributes 76
Sample: IceCrack-Special Dimensions 80
Sample: IceCrack with Sound 81
ParticlesEmitMotion 82
Attributes 82
Sample: ParticlesEmitMotion-Bubbles 87
Sample: ParticlesEmitMotion-Fire 88
ParticlesImageSwitch 89
Attributes 89
Sample: ParticlesImageSwitch 91
ParticlesInteract 92
Attributes 92
Sample: ParticlesInteract - Sweeping Rocks 96
ParticlesInteractAnimate 97
Attributes 97
Sample: ParticlesInteractAnimate - Boundary Image & Sound 107
Sample: ParticlesInteractAnimate - Collision & Sound 108
Sample: ParticlesInteractAnimate - Popcorn 109
Sample: ParticlesInteractAnimate - Flowers 110
ParticlesInteractVapor 111
Attributes 111
Sample: Sweep Away Gas Particles 114
ParticlesReset 115
Attributes 115
Sample: ParticlesReset 117
Rotate 118
Attributes 118

Sample: Rotate Image File 118
Silhouette 119
Attributes 119
Sample: Display Silhouette 119
WaterFX 120
Attributes 120
Sample: WaterFX Normal 123
Sample: WaterFX with Density Map and Light 124
Sample: WaterFX Normal with Light 125
Sample: WaterFX Normal with Light and Sound 126
Sample: WaterFX - Thick Gel 127
Sample: WaterFX - Thin Gel 128
WaterMask 129
Attributes 129
WaterOverlay 131
Attributes 131
WipeRevealExpand 133
Attributes 133
Sample: Basic WipeRevealExpand 136
Sample: WipeRevealExpand with Sound 137
WipeRevealNormal 138
Attributes 138
Sample: Basic Wipe Effect 140
WipeRevealReset 141
Attributes 141
Sample: Basic WipeRevealReset Effect 142
Play Items - Players 143
Directory 144
Attributes 144
Sample: Directory Play Item with SequentialScheduler 146
Sample: Directory Play Item with BasicTiles 146
File 147
Attributes 147
Sample: Display Media File 148
FlashFile 149
Attributes 149
Sample: Display Flash File 149
Image 150
Attributes 150
Sample: Display Image File 150
KeyedCameraViewer 151

Attributes 151
Sample: ParticlesInteractAnimate with Video 153
Sample: Bubbles in Water with Video 154
Sample: IceCrack with Sound and Video 155
Sample: ParticlesReset with Video 156
Sample: Popcorn with Video 157
Sample: Venetian Blinds with Video 158
Sample: WipeRevealExpand with Video 159
Sample: WipeRevealExpand Including Video 160
Sample: WipeRevealReset Including Video 161
MusicPlayer 162
Attributes 162
Sample: Simple MusicPlayer 163
Video 164
Attributes 164
Sample: Play Video File 165
Sample: Play Video File, Full Screen 165
Play Items - Utilities 166
AdTracker 167
Attributes 167
Sample: Track Ad 168
FrameRate 169
Attributes 169
Sample: Display Frame Rate Yellow 169
ScreenShot 170
Attributes 170
Sample: ScreenShot 171
Tracker 172
Attributes 172
Sample: Display Tracker 172
Dazzler Mouse Tracker 173
Opening the Console 174
Running the Mouse Tracker 174
Running Dazzler using Dazzler Mouse 174
Stopping the Mouse Tracker 174
Setup 175
Displaying the mouse pointer 176
Dazzler ActiveX Control 177
Using Dazzler ActiveX Control 177
Sample: ActiveX Control 178

Customizing Effects - A Tutorial 179
Getting started 180
To create a Dazzler shortcut on the desktop 180
To set up folders for new effects 180
Lesson 1: Changing the background image 182
To set up a new effect 182
To change the background image 183
To play the new effect 184
Lesson 2: Changing the behavior of an existing effect 185
To set up a new effect 185
To change the behavior of an effect 186
Lesson 3: Combining two effects 188
To set up a new effect 188
To customize the new effect 191
Lesson 4: Creating a thumbnail image of your new effect 192
Troubleshooting 193
GestureTek Contact Information 194

1 Introduction

Dazzler is GestureTek's interactive media player. Dazzler utilizes GestureTek's motion tracking technology with modules, called "play items," to create interactive and entertaining presentations.

This User Guide contains detailed descriptions and samples of each play item and a tutorial to help you customize Dazzler effects to suit yourself.

Cross-references are denoted with special page number graphics like this: .

Because Dazzler is used with several products, where file paths are indicated, the initial directories are as follows: C:\GestureTek**ProductName**...

To enter the correct path, replace 'ProductName' with the name of the product you are using (GestureFX, ScreenXtreme, Illuminate, etc.)

Supported Sound Formats

- The following sound file formats work with Dazzler:

  .wav

  .mp3

- The maximum sampling rate for .wav is 48,000 Hz.
- Stereo and mono streams are supported

Supported Image File Formats

- BMP
- DDS
- DIB
- HDR
- JPG
- PFM
- PNG
- PPM
- TGA

1.1 Copyright and Legal Matters

GestureTek and ScreenXtreme are registered trademarks of GestureTek Inc. All rights reserved.

GestureFX and Dazzler are trademarks of GestureTek Inc.

The contents of this documentation are the property of GestureTek Inc. and are for the exclusive use of registered users.

Apple and QuickTime are trademarks of Apple Inc., registered in the U.S. and other countries.

Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Flash is a registered trademark of Adobe Systems Incorporated in the United States and/or other countries.

1.2 Starting Dazzler

- On the Tracker console, with the Tracker running, click **START DAZZLER**.

Alternatively, you can select **Start>All Programs>GestureTek>ProductName>Dazzler**.

By default, Dazzler will begin in full screen mode.

If you want to make changes to the Dazzler configuration prior to starting, you need to run Dazzler Setup 3.

For more information on running a Tracker, see your Tracker's documentation.

1.3 Dazzler Setup

Accessing Dazzler Setup

If Dazzler is not running.

- Select **Start>All Programs>GestureTek>ProductName>Dazzler Setup**.

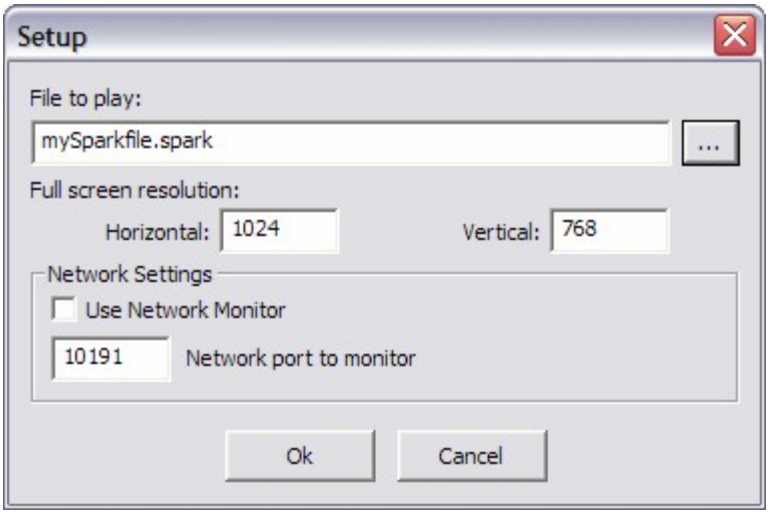
If Dazzler is running in Full Screen mode.

- Enter **Windows mode** by pressing **Alt-Enter**. Then select **Application>Setup**.

  OR

- With your mouse over the presentation area (whether or not you can see your mouse pointer), right-click the mouse and select **Setup** from the short-cut menu.

The following dialog box appears.



Dazzler Setup

Using Dazzler Setup

1. In the **File to play** field click the **Browse** button and select the Spark file that you want to run when Dazzler starts.

If the Windows folder option, "Hide Extensions for known file types" is activated, then the "Find file" pick list will not display properly. It is strongly recommended that "Hide Extensions for known file types" be deactivated.

For more information about Spark files, see Running Play Items 22.

For more information about file paths, see Understanding File Paths 26.

2. In the **Full screen resolution** section, enter the horizontal and vertical resolution of your display, if it differs from the default resolution of 1024 x 768.

   When changing "Full screen resolution", it may be necessary to close down Dazzler and restart in order for your new settings to take effect.

3. If you are running Dazzler on a separate computer from the tracker, select the **Use Network Monitor** check box.

4. In the **Network port to monitor** field, enter the port number that Dazzler will monitor, if different from the default port, 10191.

5. When you are finished making changes to **Dazzler Setup**, click **OK** to close the menu and start Dazzler.

   Dazzler will start in **Windows mode**. To change to **Full Screen mode**, wait until Dazzler is running and press **Alt-Enter**.

   If your GestureTek product includes a Control Panel, the **Active Sequence** will run each time the Control Panel is started, regardless of what file has been called for in Dazzler Setup. The file specified in Dazzler Setup will only run from Dazzler Setup.

   For more information about Dazzler's network capability, contact your GestureTek reseller.

1.4 Command Line Options

Starting the Tracker and Dazzler, using the default Spark file (Normal Method)

Starts the Tracker, waits for the autostart countdown to complete, starts camera, and then starts Dazzler. If Dazzler is shut down, the Tracker will continue to run.

From the Windows command line, enter:

```
[TrackerPath]\[Tracker.exe] -startdazzler
```

[TrackerPath] is the folder path to your Tracker program file. [Tracker.exe] is the name of your Tracker program file. (e.g. "C:\GestureTek\ProductName\ProductNametracker.exe"). Don't use quotes.

Starting the Tracker and Dazzler, using the default Spark file (QuickStart)

Starts the Tracker and immediately starts the camera and Dazzler. If Dazzler is shut down, the Tracker also shuts down.

From the Windows command line, enter:

```
[TrackerPath]\[Tracker.exe] -quickstart
```

[TrackerPath] is the folder path to your Tracker program file. [Tracker.exe] is the name of your Tracker program file. (e.g. "C:\GestureTek\ProductName\ProductNametracker.exe"). Don't use quotes.

The command option "-quickstart" is case sensitive.

Starting Dazzler with a specified Spark File

From the Windows command line, use the following string:

```
[path]\Dazzler.exe -StartupFile "[ResourcesPath]\file.spark"
```

Notes

- The Tracker needs to be running for this to work.
- This option does not change the default Spark file as specified in the Dazzler.cfg file.
- "[path]" is the path to your Dazzler System folder.
- "[ResourcesPath] is the path to your Spark file folder relative to your Resources folder. For example, if the full Windows path of the Spark file you want to run is "C:\GestureTek\ProductName\Resources\mySparks\myfile.spark", then you would specify, "mySparks\myfile.spark". Do not use a starting slash.
- The specified file must be surrounded by quotes.
- The command option "-StartupFile" is case sensitive.

Full Screen Mode vs. Windows Mode

In most cases, it is preferable to run Dazzler in **Full Screen mode**. You can toggle between **Full Screen** and **Windows mode** while Dazzler is running by pressing **Alt-Enter** on your keyboard.

1.5 Multiple Displays

Multiple displays are only possible with the **MP** version of GestureFX which supports multiple displays on systems using either one or two linked video cards.

Desktop Settings

Before configuring the settings for multiple displays, you must set the display mode for your desktop.

Horizantal Span mode (Nvidia cards): creates a single desktop across two monitors): 1 desktop/2 subdivisions for blending)

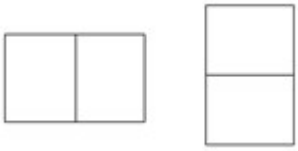
DualView Mode (ATI): creates two separate but joined desktops.: 2 desktops/2 subdivisions for blending)

Monitors must be left to right (Windows Desktop settings).

1.5.1 Setting Up Multiple Displays

There are various ways in which displays can be set up to create one large display:

- 2 displays set up horizontally or vertically, using 1 or 2 outputs on the graphics card



- 3 displays set up horizontally using 1 output on the graphics card

- 4 displays set up horizontally or in a 2 x 2 grid using 2 outputs on the graphics card

In order to get a vertically stacked display, you will need to rotate your projectors.

Current graphics drivers support a maximum displayable resolution of 4096 x 4096. (A 32bit color image of 4096 x 4096 is 64MB.)

1.5.2 Blending

Blending becomes necessary when there is more than one display. Without it, there will either be a gap between the images or a very visible double seam. The solution is to overlap the two images and then modify the pixels in the overlap region so as to make the overlap as invisible as possible.

Blending is a feature that is currently only available for GestureFX.

There are several settings which can be adjusted to obtain an almost invisible blending of displays:

Pixel Overlap:	The number of pixels by which the two displays overlap.
Blend Power:	The exponent applied to the blending function.
Blend Luminance:	The brightness level of the blending region.
Blend Gamma:	The gamma (output brightness) on each of the color channels. (Affects the blended area only.)
Color Balance:	Allows the user to adjust the overall color intensity for each display. If one display is brighter than another, you can bring its intensity down to match the other(s).
Uncropped:	Prevents the edges of the content from being cut off.
Save:	Saves the current blending settings.
Defaults:	Reloads the blending defaults.

Best results with blending are achieved when the projectors are identical, including the bulbs (bulbs in all projectors should be the same age and have the same lumens).

Make sure your projectors' RGB values are the same or close before adjusting the RGB values in the GestureFX software. (This may require using different values to get the same color.)

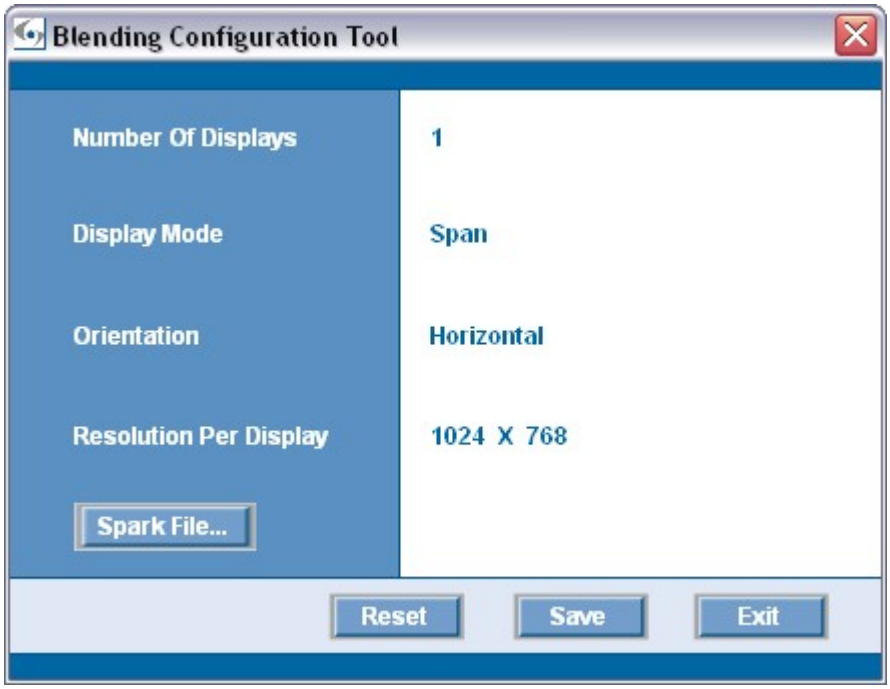
The quality of the image is also affected by the viewing distance, the size of the graphic and the distance of the projector from the display.

1.5.2.1 To configure settings for multiple displays

1. Browse to the **Tools** folder of your GestureTek product and click on the **BlendingConfigurationTool.exe** file.

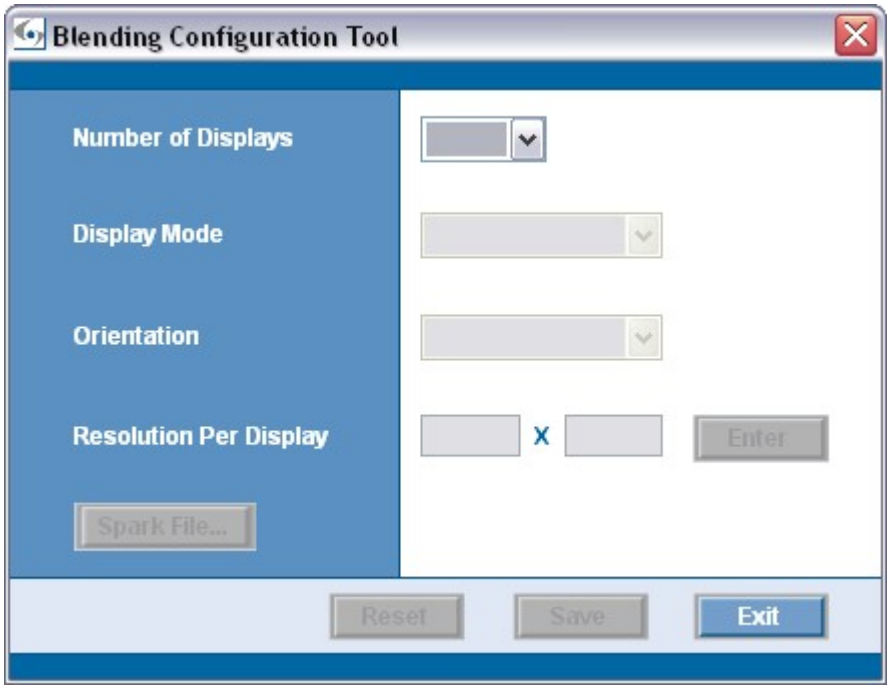
   The default location is **C:\GestureTek\ProductName\Tools**.

   The following interface appears, with settings for a single display selected.



2. To set up multiple displays, click **Reset**.

   The tool now appears with drop-down boxes from which you can select the options you want.



3. Select the number of displays you are using. The following table lists the possibilities.

Display Mode refers to the setting of your Desktop. Make sure to change this setting before using this tool.

Number of Displays	# of Outputs from Video Card	Display Mode	Orientation
2	2	Span	Horizontal
			Vertical
2	1	Span	Horizontal
2	2	Dual View	Horizontal
			Vertical
3	1	Span	Horizontal
4	2	Dual View	Horizontal
			Grid

4. Enter the **Resolution Per Display**.

Dazzler effects must be displayed in 1024 x 768 resolution. If you have created your own content, enter the appropriate resolution.

Continue: Blending Options 11

1.5.2.2 To select a blending option

1. Click **Spark File**.

   The **Select Spark File** menu appears.

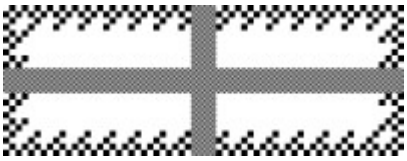
2. Click on the expand button of the display setup you are using.

3. Select a blending option.

   There are 10 blending options available, as shown below. You can select an individual blending option or **All.Spark** to cycle through all the options.

Band.spark

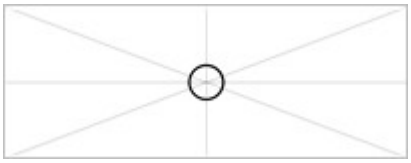
BlackBox.spark



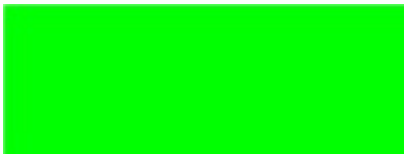
BlackWhiteGrid.spark

Blue.spark

ColorAlignment.spark



DiagonalLines.spark



Green.spark

Grey.spark

Red.spark

White.spark

Starting with the **Color Alignment** blending interface is recommended, as it assists in aligning the displays with the **Pixel Overlap** setting.

4. Click **Save, OK** and then **Exit**.

5. Run **Dazzler**.

The blending interface you chose appears. This consists of a background image with two types of blending boxes, the **Universal Blending Interface** (the larger box) and a **Color Balance Interface**. Each display has its own **Color Balance Interface**. The image below is the **ColorAlignment.spark** interface.

Continue: Adjusting Universal Blending Interface settings 14

1.5.2.3 To adjust Universal Blending Interface settings

1. Tab to the **Universal Blending Interface**. See figure below.

2. Use the down arrow to highlight the **Pixel Overlap** setting and the left and right arrows to adjust the value of the setting. The goal is to line up the checkered areas, the colored bars and the circle (in the **Color Alignment** blending interface) so that the overlap area appears as one seamless, sharp image. The minimum recommended **Pixel Overlap** is 96.

   Page Up and **Page Down** adjust the values by 0.10. Left and right arrows adjust the values by 0.01.

3. Adjust **Power** to increase or decrease the fading out effect of the overlap area of the image. A value that is too high will leave a very bright or very dark line in the center of the overlap area. A value that is too low will leave a wider, though less bright or dark, area in the center. The minimum value is 0.0, maximum is 10.0. Default value is 1.0.

4. Adjust **Luminance** only if there is a very pronounced brightness or darkness in the overlap area. Values lower than 0.5 will darken the area. Values higher than 0.5 will brighten it. Default value is 0.50 and would rarely need to be changed.

5. Adjust the **Red**, **Green** and **Blue Gamma** settings so that the blended area of the image appears to have the same level of brightness as the rest of the image.

   Each color channel (red, green and blue) may be adjusted individually by using the up/down arrow keys to select a particular channel and then holding down the **CTRL** key while using the left/right arrow key. Using the left/right arrow keys alone will change all channels together.

   Each color channel has a value between 0.0 and 1.0. A value of 0.0 has no intensity, meaning there would be no color at all from that channel. 1.0 is full intensity of that channel. If all channels are set to 0.0, the screen will be black. Default is 1.0 for all channels.

6. Once you are satisfied with the **Universal Blending** settings, check your image by using the other blending interfaces. Problem areas will show up better on some interfaces than others.

7. Set **Uncropped** to **True** to prevent the edges of the content from being cut off.

8. Highlight **Save** and press **Enter** to save the settings.

The word **Succeeded** appears when the settings are saved.

To restore the default settings, highlight **Default** and press **Enter**.

Continue: Color Balance settings 15

1.5.2.4 To adjust Color Balance Interface settings

1. Press the tab key to move from the **Universal Blending Interface** to the **Color Balance Interface** (shown below) of each display.

2. Adjust the **Red**, **Green** and **Blue** channels so that the image in each display has the same intensity of color. Use the up/down arrows to select a channel and then hold down the **CTRL** key while using the left/right arrows to adjust the values. Test your settings on different blending interfaces to make sure they are accurate.

Each color channel has a value between 0.0 and 1.0. A value of 0.0 is no intensity, meaning there would be no color at all from that channel. 1.0 is full intensity of that channel. If all channels are set to 0.0, the screen will be black. Default is 1.0 for all channels.

3. When you are satisfied with the **Color Balance** settings, tab back to the **Universal Blending Interface**, select **Save** and press **Enter** to save these settings.

4. On the Tracker console, with the Tracker running, click **START DAZZLER**.

1.6 Dazzler Logs

Dazzler logs provide important information about the running of Dazzler, including both the Dazzler player itself (useful to system engineers) and the running of Spark files (useful to Spark file editors).

Enable/disable Dazzler logs

To enable or disable Dazzler logs, change the value of the "**LogEnabled**" attribute in the **DazzlerKernel.cfg** file (the file is located in the Config folder). Setting the value to "True" enables logging. Setting it to "False" disables logging.

When logging is enabled, a log file for each session will be written to the **Logs** folder.

The **Logs** folder must exist for log files to be created. Removing the **Logs** folder effectively disables logging. This is an intentional feature.

Controlling Dazzler Log Detail

The amount of detail reported in the Dazzler logs can be controlled via the **LogDetailLevel** attribute in the **DazzlerKernel.cfg** file (the file is located in the **Config** folder). The available values are 0-9, where "0" provides the least amount of detail, and "9" provides the most. The default value is "1", which is sufficient in most cases.

1.7 Exiting Dazzler

Full Screen Mode

To exit Dazzler in **Full Screen Mode**, press your keyboard's **Esc** key or **Alt-F4**.

Windows Mode

To exit Dazzler in **Windows** mode, press your keyboard's **Esc** key or **Alt-F4**. Alternatively, use the menu to select **Application**>**Exit**.

2 About Play Items

Play items are preinstalled modules, which control what is viewed in a Dazzler presentation. Several play items are included with your installation. Upgraded and new play items will be sent to users as they become available, depending on your current licensing agreement. Customized play items are also available.

Your installation may or may not include all play items listed. If you are interested in acquiring additional play items, contact support@gesturetek.com.

There are four types of Dazzler play items. Understanding these types will help you use them more effectively.

Controller play items provide system control. Controller play items allow you to do such things as schedule and sequence certain play items and automatically shut down Dazzler.

Effect play items provide visual and interactive effects. Dazzler contains a suite of 25 standard effects.

Player play items display or run one or more media files, including images, videos, Adobe Flash®, and sound.

Utility play items are tools to help developers create effective Spark files.

Each play item includes a description as well as explanations of all available attributes for that particular play item and where applicable, their associated value types, value ranges and default values. One or more code samples for each play item are also provided.

This section provides information to help you understand play items, as follows:

Play Item Attributes 18
Setting Color Values 19
The Duration Attribute 20
Running Play Items 22
Spark Files 22
Nested Play Items 25
Understanding File Paths 26
Using Sample Code 27
Dazzler Demos 29
Hot Keys 29
Alphabetical List of Play Items 30

2.1 Play Item Attributes

Attribute Types

Most play items include one or more attributes that control various aspects of that play item. There are four types of attributes.

- **Required**: A play item attribute that must be included. If a **Required** attribute is not specified, the play item will function incorrectly or not at all.
- **Default**: A play item attribute that has a default value. If a **Default** attribute is not specified, the default value for that attribute will be used.
- **Optional**: **Optional** attributes only have an effect on a play item when listed.
- **Dependent**: **Dependent** attributes are similar to **Default** attributes, except that they are only in effect when the **Optional** attributes they are dependent on are listed. In that case, if the **Dependent** attribute is not listed, its default value will be used. See actual play items for more information.

Attribute Value Types

Attribute values must be specified in a format in keeping with its particular type.

- **Boolean**: Logical values as in "True" or "False", "Yes" or "No", "On" or "Off", "1" or "0".

  Where Boolean values are specified, any of the above logical choices are allowed.

- **Decimal**: Decimal numbers.
- **Enumerated**: Text value based on a list of options.
- **FilePath**: Requires a folder or file name location. All file and folder names must be comprised of letters, numbers, underscores and dashes only.
- **Integer**: Whole numbers. No decimals.
- **String**: A series of characters, such as "ABCD"

Working with Attribute Values

It is important to remember that Dazzler play items are affected by various combinations of attributes and their values. Some attributes affect a play item in an absolute way (e.g. a particle moving across the screen in a particular direction at a particular speed), while others do so in a relative way (a more watery or less watery appearance). Each attribute is described under each play item.

Several play item attributes allow for very high or infinite values. Even though these high values are supported by the software, they are dependant upon system resources such as memory, processor speed, graphics card capabilities, etc. or simple practicalities such as screen clutter, conflicting or too many sounds, etc. One should also be aware of how various attributes work in combination with other attributes. Using certain attributes may affect the running of other attributes and/or certain attribute settings may reduce the limitations of other attributes.

The overall behavior of play item attributes is dependant on system resources and the type of hardware used.

2.2 Setting Color Values

Certain play items include color setting attributes. Color values are defined by specifying red, green and blue components. Each component is in the range of 0.0 to 1.0. When using a color attribute, all components must be assigned a value. Default color values depend on the specific play item.

Example color values:

Black: "0.0, 0.0, 0.0"

Blue: "0.0, 0.0, 1.0"

Green: "0.0, 1.0, 0.0"

Magenta: "1.0, 0.0,1.0"

Orange: "1.0, 0.5, 0.1"

Red: "1.0, 0.0, 0.0"

White: "1.0, 1.0, 1.0"

Yellow: "1.0, 1.0, 0.0

Alpha Channel Support

In cases where an alpha channel (transparency) is supported, a fourth component is required, which specifies the percentage of opaqueness, where "0.0" is fully transparent and "1.0" is completely opaque. For example, 60 percent opaque red is specified by "1.0, 0.0, 0.0, 0.6".

2.3 The Duration Attribute

The **Duration** attribute is a special attribute that is required when a play item is an immediate nested play item (child element) of a **Scheduler** play item (AttractLoopScheduler 32, RandomScheduler 36, SelectionScheduler 38, or SequentialScheduler 39). When a **Duration** attribute value is specified, it sets, in seconds, the duration of that particular play item.

Notes

- For examples of the correct and incorrect use of the **Duration** attribute, see Using the Duration attribute 20.
- For more information on nested play items, see Nested Play Items 25.
- When transition play items, such as AlphaTransition 49, IceCrack 76, BasicTiles 51, WipeRevealExpand 133, and WipeRevealNormal 138, are within a SequentialScheduler 39 and their **Loop** attribute is set to "False", that play item will, upon its final transition, terminate, thus allowing the next play item within the SequentialScheduler 39 to start, regardless of the value of the **Duration** attribute.

  For example, if a BasicTiles 51 effect inside a SequentialScheduler 39 has its **Duration** value set to 30 seconds and its **Loop** value set to "False", but it only takes 12 seconds to flip through and display all of its child play items, then the BasicTiles 51 effect will end after 12 seconds and the SequentialScheduler 39 will immediately move on to the next play item in the list.
- Special uses of the **Duration** attribute will be discussed as needed under relevant play items.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 0.0

2.3.1 Using the Duration attribute

The **Duration** attribute is required when using a **Scheduler** play item. It must be placed as an attribute of all the **Scheduler's** first-level child play items. It should not be used anywhere else.

Examples

The following are correct and incorrect examples of using the **Duration** attribute within a **Scheduler**. They use a variety of generic terms for illustration purposes only. They are not working samples. "n" refers to any number of seconds.

The following two examples are correct:

```
<SequentialScheduler>
  <PlayItem
    Duration="n"
    OtherAttribute="value"
    OtherAttribute="value"
  >
      <PlayItem Attribute="value"/>
  </PlayItem>
</SequentialScheduler>

<SequentialScheduler>
  <PlayItem Attribute="value" Duration="n" />
  <PlayItem Attribute="value" Duration="n" />
</SequentialScheduler>
```

The following two examples are not correct

```
<SequentialScheduler>
  <PlayItem
    OtherAttribute="value"
    OtherAttribute="value"
  >
      <PlayItem Attribute="value" Duration="n"/>
  </PlayItem>
</SequentialScheduler>

<SequentialScheduler Duration="n">
  <PlayItem Attribute="value" />
  <PlayItem Attribute="value" />
</SequentialScheduler>
```

2.4 Running Play Items

Play items are run by creating XML-based text files, called Spark files. Spark files must end with the extension ".spark" and be stored in the **Resources** folder and its subfolders.

Spark files list one or more play items and their attribute settings. Your installation includes a library of sample Spark files that you may use or edit (see Using Sample Code 27).

Spark file errors are written to Dazzler Logs. For more information on log files, see Dazzler Logs 16.

How do Spark files work?

At start up Dazzler reads the default Spark file specified in the Dazzler configuration (to change the default Spark file, see Dazzler Setup 3). The play items will run according to their specified settings and in the order listed or until Dazzler is terminated. Unless otherwise specified, the current Spark file will repeat indefinitely or until Dazzler is shut down.

2.5 Spark Files

Spark File Structure

Before learning to create and edit Spark files, it is essential to understand their structure.

Spark files follow the XML 1.0 specification. While knowledge of XML is not absolutely necessary in order to create and edit Spark files, basic familiarity with XML is helpful. An XML Tutorial is available at:http://www.w3schools.com/xml/default.asp

Spark files use custom XML tags and a layout specific to Dazzler. Any errors resulting from bad XML code or incompatibility with Dazzler custom tags and layout will be reported in the Dazzler logs. For more information on log files, see Dazzler Logs 16.

Spark file names must only be comprised of letters, numbers, underscores and dashes.

Spark File Layout

The following is a basic layout that applies to all Spark files. The actual attributes for each play item, the number of attributes and their values, depend on the particular play item. Explanation of this layout follows. The layout of the Spark file below, including line breaks and indentation, is for purposes of clarity only. The layout of the actual file may not look exactly the same as this. Line numbers are provided for convenience only. ***Do not type line numbers into your Spark files!***

```
 1  <?xml version="1.0"?>
 2
 3 <!-- My Play Item Name -->
 4   <PlayItem1
 5      Attribute1 = "value"
 6      Attribute2 = "value"
 7      Attribute3 = "value"
 8      >
 9        <ChildPlayItem1 Attribute1="value" />
10    </PlayItem1>
```

Explanation of Spark file layout

The following is a line-by-line explanation of the above Spark file layout.

```
1  <?xml version="1.0"?>
```

Spark files must begin with the above XML version tag as the very first line.

The play item samples do not contain this line, since they are samples of play items themselves and not complete Spark files. All Spark files require an occurrence of the XML version tag at the very beginning of the file.

```
2
```

For the sake of good, clean layout, it is recommended to leave the second line blank.

```
3 <!-- My Play Item Name -->
```

It is recommended that an XML comment be used to describe the contents of the current Spark file.

```
4  <PlayItem1
```

The first or main play item is specified next. Play item names must be spelled correctly with no spaces.

Play item name element tags are case sensitive.

There are two styles of play item name elements

Tag blocks. These play item name elements have opening and closing XML tags as follows:

```
<PlayItemName></PlayItemName>
```

The tag pair must be spelled correctly and identically. Any attributes are placed inside the opening tag. The sample layout being discussed here is an example of a tag block.

Self-terminating tags. These play items only have one XML element as follows:

```
<PlayItemName />
```

Any attributes are placed inside the tag.

```
5    Attribute1 = "value"
6    Attribute2 = "value"
7    Attribute3 = "value"
```

The attributes, which control the settings for the particular play item, are listed next. It is recommended that play items with only one or two attributes should be listed on one line, while play items with more than two attributes should be listed one per line. Each play item has its own set of available attributes. Attributes must be spelled correctly (they are not case sensitive). Attributes are followed by an equals sign (=) and a relevant value enclosed inside a pair of double quotes (" "). For more information, see the Understanding Play Item Attributes 18 section as well as the play items themselves.

```
8    >
```

When more than one attribute is listed for a particular play item, it is recommended to place the closing bracket of the opening tag on its own line.

```
 9       <ChildPlayItem1 Attribute1="value" />
```

Many play items support the listing of one or more child play items. In most cases the parent play item will affect the child play item in some way. Child play items are listed in the same manner as parent play items. Child play items can have their own child play items. This is an example of a self-terminating tag, but tag block play items can be used for child play items as well.

```
10   </PlayItem1>
```

Since this play item is a tag block, it requires a closing tag. For the purposes of good layout, we recommend listing it on its own line.

For actual working examples of Spark file contents, see the sample code provided with each play item.

2.6 Nested Play Items

While certain play items run on their own, many require the nesting of at least one other play item. For instance, a water effect usually applies a liquid look to another play item, such as an **Image** play item or even another effect play item.

Since play items are XML elements, the technical way to refer to the nesting of play items is in terms of parents and children. In the following example a **WaterFX** play item is applying a liquid effect to an **Image** play item. The **WaterFX** play item is the parent element and the **Image** play item is the child element.

```
<WaterFX
  Density="0.98"
  Speed="20.0"
  RefractionFactor="1.0"
  Height="0.05"
  InteractionMethod="Solid"
  >
    <Image Name="Path\file.ext"/>
</WaterFX>
```

Some play items allow for a sequence of two or more child play items. Some play items, when used as a child play item, may also have their own child play items. Some play items don't allow child play items at all.

More information regarding the nesting of play items is found in the play item descriptions as well as in the sample code.

2.7 Understanding File Paths

The Resource folder

All Spark files and resource files (images, sounds, videos, etc.) must be stored in the **Resources** folder or its subfolders. You can create whatever folders you wish as long as they are in the **Resources** folder.

All Spark file and resource file (images, sounds, videos, etc.) names must be comprised of letters, numbers, underscores and dashes only.

Specifying file paths in Spark files

There are two methods when specifying file paths in Spark files:

1. Relative to the Resources folder

Using this method, a file whose full path is:

c:\GestureTek\ProductName\Resources\Presentations\Summer\Images\myimage.jpg

would be specified in your Spark file as:

```
Presentations\Summer\Images\myimage.jpg
```

When using this method, do not place a slash at the beginning of your path.

2. Relative to the folder your Spark file is in

Using this method, if your Spark file is in the folder:

c:\GestureTek\ProductName\Resources\Presentations\Summer

and you want to specify a file whose full path is:

c:\GestureTek\ProductName\Resources\Presentations\Summer\Images\myimage.jpg

you would specify that file in your Spark file as:

```
.\Images\myimage.jpg
```

Notes

- Spark files do not support the referencing of full path names. In other words, you would never specify the following in a Spark file:

  c:\GestureTek\ProductName\Resources\Presentations\Summer\Images\myimage.jpg

- Spark files do not support references to parent folders ("..\"). Therefore if a file is above the folder structure of a particular Spark file's location, then the path specified must be relative to the **Resources** folder (as in method 1 above).

2.8 Using Sample Code

The behavior of the samples included with your installation is dependant on your system resources, including the type of hardware used.

The sample code examples in this documentation are taken from the sample Spark files found in your **Resources\Manual_Samples** folder.

You may use the Sample Code examples as the basis of your own Spark files. The supporting files (images, sounds, etc.) are located in the subfolders of the **Resources\Manual_Samples** folder. ScreenXtreme manual samples, using the **KeyedCameraViewer** play item, are located in **Resources\Manual_Samples** folders. When using the sample code for your own use, make sure you adjust the file and folder paths accordingly. For more information on how to specify file and folder paths, see Understanding File Paths 26.

ScreenXtreme Users! This Dazzler User Guide includes many samples that also work with GestureFX. GestureFX does not use the **KeyedCameraViewer** play item and thus does not display the camera's video image as ScreenXtreme does. Special samples demonstrating how to use the **KeyedCameraViewer** with several other play items are listed in the KeyedCameraViewer 151 section. The other samples will work with ScreenXtreme, but will not display the video image unless the **KeyedCameraViewer** is added.

Included in the **Resources\Manual_Samples** folder is an additional Spark file called **All.spark**, that plays almost all of the sample Spark files.

It is assumed that you have at least a basic understanding of how to configure Dazzler and how to create and edit Spark files before using the sample code.

Sample Code Format

All Spark files follow standard XML specifications.

All Spark files must begin with the following XML version header:

```
<?xml version="1.0"?>
```

While XML elements may by written on single or multiple lines, the following layout is preferred. Only play items with one or two attributes should be written as single line elements.

```
<PlayItem name="path\image.ext" />
```

XML elements with three or more attributes should be written on multiple lines with the closing element bracket on its own line.

```
<PlayItemName
  Attribute1="value"
  Attribute2="value"
  Attribute3="value"
  >
```

Tag blocks require a separate terminating tag:

```
<PlayItemName
  Attribute1="value"
  Attribute2="value"
  Attribute3="value"
  >
    Child Play Item1
    Child Play Item2
    Child Play Item3
</PlayItemName>
```

Self-terminating tags require a closing slash to be placed before closing bracket.

```
<PlayItem name="Path\image.ext" />
```

Play item names (XML element names) must be spelled exactly as shown in the Dazzler play items documentation. Play item names are case sensitive.

2.9 Dazzler Demos

Your installation includes the Spark files used in this User Guide as well as several other demo Spark files.

Demos

Effect Play Items

A demonstration of **Effect** play items may be viewed by running the **root.spark** file located in your **Resources\Demos** folder.

ScreenXtreme Demos

A special set of demos for ScreenXtreme is located in your **Resources\Demos\ScreenXtreme** folder.

Manual Samples

For information about the samples included with this documentation, see Using Sample Code 27.

2.10 Hot Keys

Dazzler features the following shortcut key combinations:

Ctrl-F: Toggles the display of the current frame rate. For more information see FrameRate 169.

Ctrl-I: Toggles the display of monitor identification. Useful when using multiple displays.

Ctrl-T: Toggles the display of the Tracker outline. For more information see Tracker 172.

2.11 Play Items - Alphabetically

AdTracker 167

AlphaMask 45

AlphaOverlay 47

AlphaTransition 49

AttractLoopScheduler 32

BasicTiles 51

ChildPlayer 33

ColourKeying 59

Directory 144

Exit 33

File 147

FishSwarm 62

FlashFile 149

FluidDynamics 65

FrameRate 169

GenericSwarm 67

Greyscale 74

IceCrack 76

Image 150

Include 34

KeyedCameraViewer 151

MusicPlayer 162

ParticlesEmitMotion 82

ParticlesImageSwitch 89

ParticlesInteract 92

ParticlesInteractAnimate 97

ParticlesInteractVapor 111

ParticlesReset 115

RandomScheduler 36

Rotate 118

ScreenShot 170

SelectionScheduler 38

SequentialScheduler 39

Silhouette 119

SparkFile 41

Tracker 172

Video 164

WaitForMotion 42

WaterFX 120

WaterMask 129

WaterOverlay 131

WipeRevealExpand 133

WipeRevealNormal 138

WipeRevealReset 141

3 Play Items - Controllers

Controller play items provide system control and allow you to do such things as schedule and sequence certain play items and automatically shut down Dazzler.

AttractLoopScheduler 32
ChildPlayer 33
Exit 33
Include 34
RandomScheduler 36
SelectionScheduler 38
SequentialScheduler 39
SparkFile 41
WaitForMotion 42

3.1 AttractLoopScheduler

This is a **Controller** play item that allows for two different **Child Play** items to run: the first, when no motion is detected; the second, only upon motion detection.

Notes

- The play item that will run when there is no motion detected must be listed first.
- As soon as motion is detected, the second play item will run.
- If motion is detected at the same time as the **AttractLoopScheduler** starts, then the second item will display right away.
- Nested play items are permitted within either child play item.
- It is not recommended to place an **AttractLoopScheduler** play item inside an **Include** play item as it will return unexpected results.

3.1.1 Attributes

Duration 20 *(Default)*

Specifies in seconds how long the second play item (motion detected) will continue to run once motion is no longer detected.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 0.0

3.1.2 Sample: Simple AttractLoopScheduler

Creates a simple attract loop using two **Image** play items. The first item will display only when there is no motion detected. When motion is detected the second item will display until no motion is detected for 5 seconds at which time the first item will display.

AttractLoopScheduler\root.spark

```
<AdTracker AdName="Manual_Samples\AttractLoopScheduler">
  <AttractLoopScheduler duration="5">
    <Image Name=".\Images\red_curtain.jpg"/>
    <Image Name=".\Images\crowd.jpg"/>
  </AttractLoopScheduler>
</AdTracker>
```

3.2 ChildPlayer

This is a **Controller** play item that combines two or more play items and treats them as one. Useful for testing and debugging. It is also required for the **AlphaOverlay** and **ColorKeying Shader Filters**.

3.2.1 Attributes

None, except for a Duration attribute 20 when a **ChildPlayer** is inside a SequentialScheduler 39.

3.2.2 Samples

See the Tracker 172 and FrameRate 169 play items for more information and sample code.

3.3 Exit

This is a **Controller** play item that causes Dazzler to completely shut down.

It is recommended that the **Exit** play item only be used as a child play item within a **Scheduler** play item.

The **Exit** play item will take precedence over any other instruction. If Dazzler encounters the **Exit** play item at any time, it will immediately shut down.

3.3.1 Attributes

None

3.3.2 Sample: Exit

Displays three **Image** play items for ten seconds each before shutting down Dazzler.

Exit\root.spark

```
<AdTracker AdName="Manual_Samples\Exit">
  <SequentialScheduler>
    <Image Name=".\Images\clown_fish.jpg" Duration="5" />
    <Image Name=".\Images\moon.jpg" Duration="5" />
    <Image Name=".\Images\pavement.jpg" Duration="5" />
    <Exit />
  </SequentialScheduler>
</AdTracker>
```

3.4 Include

This is a **Controller** play item that inserts the contents of one Spark file into another. The included Spark file may also contain **Include** play items.

If the **Include** play item is a child play item within a SequentialScheduler 39, a Duration 20 attribute should be set for the **Include** play item. If not, then the SequentialScheduler 39 will attempt to use the Duration 20 setting of the root element of the specified Spark file. If no Duration 20 value is specified in either place, Dazzler will most likely return undesirable results.

Samples

In addition to the sample files included in this section, see also the **all.spark** files in the **Manual_Samples**\ folder.

3.4.1 Attributes

Name *(Required)*

Specifies the file path of the Spark file to include.

Spark file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

3.4.2 Sample: Include within SequentialScheduler

The following sample contains a SequentialScheduler 39 play item that lists two **Include** play items:

Include1\root.spark

```
<AdTracker AdName="Manual_Samples\Include1">
  <SequentialScheduler>
    <Include name=".\myimages1.spark" duration="15" />
    <Include name=".\myimages2.spark" duration="15" />
  </SequentialScheduler>
</AdTracker>
```

The **Include** play items in the above example refer to the following two Spark files:

Include1\myimages1.spark

```
<!-- FILE #1 TO DEMONSTRATE INCLUDE PLAY ITEM -->
<SequentialScheduler>
  <Image Name=".\Images\moon.jpg" Duration="5" />
  <Image Name=".\Images\tower.jpg" Duration="5" />
</SequentialScheduler>
```

Include1\myimages2.spark

```
<!-- FILE #2 TO DEMONSTRATE INCLUDE PLAY ITEM -->
<SequentialScheduler>
  <Image Name=".\Images\hawaii.jpg" Duration="5" />
  <Image Name=".\Images\crowd.jpg" Duration="5" />
</SequentialScheduler>
```

The **root.spark** file instructs the contents of **myimages1.spark** to run for 15 seconds, after which the contents of **myimages2.spark** will also run for 15 seconds. Then **myimages1.spark** will begin to run again, and so on.

The items specified in the included Spark files are each listed within their own SequentialScheduler 39 play items and have their own durations 20. They will run for those durations, looping in sequence until the duration specified in each **Include** play item listed in the **root.spark** file is complete.

3.4.3 Sample: Simple Include

Runs the Spark file specified in the **Name** attribute.

Include2\root.spark

```
<AdTracker AdName="Manual_Samples\Include2">
  <Include name=".\Water\root.spark" />
</AdTracker>
```

3.5 RandomScheduler

This is a **Controller** play item that will randomly play one or more child play items from a series. Each child item will run for the duration specified within that item's **Duration** 20 attribute.

It is essential that you understand the **Duration** 20 attribute before using the **RandomScheduler**.

3.5.1 Attributes

NumberOfChildrenToSchedule *(Default)*

Specifies the number of items to play. If you specify a value greater than the number of items available, items will be repeated. The default value of "0" will cause all child items to play.

Value Specifications: Type: Integer; Range: 0 and up; Default: 0

NumberOfIterations *(Default)*

Determines how many times the scheduler will play the randomized child items. The default setting of "0" causes the items to play indefinitely.

Value Specifications: Type: Integer; Range: 0 and up; Default: 0

RandomType *(Default)*

Sets the random selection method. "Pure" (the default setting) randomly selects from all available child items each time an item is set to play. "NoRepeats" will not allow any item to repeat until all items have played.

Value Specifications: Type: Enumerated; Options: "Pure" or "NoRepeats"; Default: "Pure"

ReselectEveryIteration *(Default)*

If set to "True", then the scheduler will create a fresh randomization upon each iteration. If set to "False" (the default setting), then the scheduler will run the same order upon every iteration. This attribute only has an effect if the **NumberOfIterations** attribute is not set to "1".

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

3.5.2 Sample: Play Random Images

RandomScheduler\root.spark

Random playing of all images in a folder for an indefinite number of iterations. All images will play in each iteration. A different order of images will be set each time.

```
<AdTracker AdName="Manual_Samples\RandomScheduler">
  <RandomScheduler
    RandomType="NoRepeats"
    NumberOfChildrenToSchedule="0"
    NumberOfIterations="0"
    ReselectEveryIteration="True"
    >
      <Directory Name=".\Images" Filter="*.*" Duration="5" />
  </RandomScheduler>
</AdTracker>
```

3.6 SelectionScheduler

This is a **Controller** play item that will play one specified child play item from a series. The selected child item will run for the duration specified within that item's Duration 20 attribute.

It is essential that you understand the Duration 20 attribute before using the **SelectionScheduler**.

3.6.1 Attributes

ItemToPlay *(Default)*

Specifies which child play item to play.

Value Specifications: Type: Integer; Range: 0 and up; Default: 0

The first item is designated as "0", the second item as "1", etc.

NumberOfIterations *(Default)*

Sets how many times the scheduler will loop over its series of child play items. The default setting of "0" will cause it to loop indefinitely.

Value Specifications: Type: Integer; Range: 0 and up; Default: 0

Notes

1. The selected child play item will run the specified **NumberOfIterations** times the number of seconds specified in the Duration 20 attribute.

2. When the last iteration plays, it will no longer be displayed, but Dazzler will continue to run.

3.6.2 Sample: Play Second Child Play Item

Plays only the second child play item for five seconds.

SelectionScheduler.spark

```
<AdTracker AdName="Manual_Samples\SelectionScheduler">
  <SelectionScheduler
    ItemToPlay="1"
    NumberOfIterations="1"
    >
      <Image Name=".\Images\clown_fish.jpg" Duration="5" />
      <Image Name=".\Images\earth.jpg" Duration="5" />
      <Image Name=".\Images\logo.jpg"Duration="5" />
  </SelectionScheduler>
</AdTracker>
```

3.7 SequentialScheduler

This is a **Controller** play item that runs a series of child play items in sequence in the order listed. Each child item will run for the duration specified within that item's Duration 20 attribute.

It is essential that you understand the Duration 20 attribute before using the **SequentialScheduler**.

3.7.1 Attributes

NumberOfIterations (Default*)*

Sets the number of times that the scheduler will loop before terminating. The default setting of "0" causes the scheduler to loop indefinitely.

Value Specifications: Type: Integer; Range: 0 and up; Default: 0

Notes

- This play item requires the addition of a Duration 20 attribute in each of its child play items.
- The Duration 20 attribute of a child play item has precedence over anything else in that play item. Once a duration is complete, the **SequentialScheduler** will cause that item to terminate and the next item to begin.

3.7.2 Sample:IncludewithinSequentialScheduler

The following sample contains a SequentialScheduler 39 play item that lists two **Include** play items:

Include1\root.spark

```
<AdTracker AdName="Manual_Samples\SequentialScheduler">
  <SequentialScheduler>
    <Include name=".\myimages1.spark" duration="15" />
    <Include name=".\myimages2.spark" duration="15" />
  </SequentialScheduler>
</AdTracker>
```

The **Include** play items in the above example refer to the following two Spark files:

Include1\myimages1.spark

```
<!-- FILE #1 TO DEMONSTRATE INCLUDE PLAY ITEM -->
<SequentialScheduler>
  <Image Name=".\Images\moon.jpg" Duration="5" />
  <Image Name=".\Images\tower.jpg" Duration="5" />
</SequentialScheduler>
```

Include1\myimages2.spark

```
<!-- FILE #2 TO DEMONSTRATE INCLUDE PLAY ITEM -->
<SequentialScheduler>
  <Image Name=".\Images\hawaii.jpg" Duration="5" />
  <Image Name=".\Images\crowd.jpg" Duration="5" />
</SequentialScheduler>
```

The **root.spark** file instructs the contents of **myimages1.spark** to run for 15 seconds, after which the contents of **myimages2.spark** will also run for 15 seconds. Then **myimages1.spark** will begin to run again, and so on.

The items specified in the included Spark files are listed each within their own SequentialScheduler 39 play items and have their own durations. They will run for those durations, looping in sequence until the duration 20 specified in each **Include** play item listed in the **root.spark** file is complete.

3.7.3 Sample: Simple SequentialScheduler

This sample creates a fairly simple sequence of three **Image** play items running for five seconds each. After the third item is complete, the sequence will begin again with the first one.

SequentialScheduler\root.spark

```
<AdTracker AdName="Manual_Samples\SequentialScheduler">
  <SequentialScheduler>
    <Image Name=".\Images\crowd.jpg" Duration="5" />
    <Image Name=".\Images\moon.jpg" Duration="5" />
    <Image Name=".\Images\hawaii.jpg" Duration="5" />
  </SequentialScheduler>
</AdTracker>
```

3.8 SparkFile

This is a **Controller** play item that allows a Spark file to be called from another Spark file or from a Flash® file. Unlike the Include 34 play item, **SparkFile** gives complete control over to the new Spark file. Any subsequent instructions in the Spark file or the Flash file that calls the **SparkFile** play item will be ignored.

3.8.1 Attributes

Name (Required)

Specifies the path to the Spark file to run.

Spark file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Flash® files must be registered using the GestureTek Flash Register security application. The GestureTek Flash SDK outlines a technique for adding interactions between the user and Flash® objects.

3.8.2 Sample: Calling Spark File from a Spark File

The following sample calls a Spark file from another Spark file from within a SequentialScheduler 39. After playing an image for five seconds, Dazzler will run **file2.spark**. The called Spark file is similar to the first one. It too plays an image for five seconds before calling the first Spark file. This would continue until Dazzler is shut down.

sparkFile\file1.spark & sparkFile\file2.spark

```
<AdTracker AdName="Manual_Samples\SparkFile">
  <SequentialScheduler>
    <Image Name=".\Images\moon.jpg" Duration="5" />
   <SparkFile Name=".\file2.spark" />
  </SequentialScheduler>
</AdTracker>

<!-- SPARKFILE SIMPLE DEMO - FILE2.SPARK -->
<SequentialScheduler>
  <Image Name=".\Images\clown_fish.jpg" Duration="5" />
  <SparkFile Name=".\file1.spark" />
</SequentialScheduler>
```

3.9 WaitForMotion

This is a **Controller** play item that will play its child item until it detects user interaction or until it times out.

Notes

- **WaitForMotion** must be placed inside a SequentialScheduler 39.
- **WaitForMotion** does not support the Duration 20 attribute.

3.9.1 Attributes

PointsThreshold *(Default)*

Sets how sensitive **WaitForMotion** is to motion. The higher the number, the more user interaction is needed before **WaitForMotion** will terminate.

Value Specifications: Type: Integer; Range: Greater than 0; Default: 100

For best results, use values between 50 and 250.

Timeout *(Default)*

Sets the amount of time, in seconds, before the **TimeoutRule** is invoked.

Setting **Timeout** to "0.0" is a special value that deactivates **TimeoutRule**. A timeout will not occur when **Timeout** is set to "0.0".

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 0.0

TimeoutRule *(Default)*

Specifies one of three actions that will be taken if a timeout occurs based on the value of the **Timeout** attribute. The three actions are:

1. **StopOnTimeout**: When the **Timeout** value is met, **WaitForMotion** will terminate, allowing the SequentialScheduler 39 to run the next play item. This action provides the same results as detecting motion.
2. **ExitOnTimeout**: If the **Timeout** value is met before motion is detected, Dazzler will shut down completely.
3. **PlaySparkOnTimeout**: If the **Timeout** value is met before motion is detected, the Spark file specified in the **TimeOutSparkFile** attribute will run.

Value Specifications: Type: Enumerated; Range: "StopOnTimeout", "ExitOnTimeout", or "PlaySparkOnTimeout"; Default: "StopOnTimeout"

TimeoutSparkFile *(Dependent)*

If the **TimeoutRule** attribute is set to **PlaySparkOnTimeout**, specifies the Spark file to play when a timeout occurs.

Dazzler will give complete control to the specified Spark file once it begins to play.

Value Specifications: Type: FilePath

3.9.2 Sample: Simple WaitForMotion

Plays one image while no motion is detected. Once motion is detected, will play another image for five seconds before returning to the first image and waiting for motion again.

waitformotion1\root.spark

```
<AdTracker AdName="Manual_Samples\WaitForMotion1">
  <SequentialScheduler>
    <WaitForMotion
      PointsThreshold="200"
      TimeoutRule="StopOnTimeout"
      Timeout="0"
      >
        <Image Name=".\Images\earth.jpg" />
    </WaitForMotion>
    <Image Name=".\Images\moon.jpg" Duration="5"/>
  </SequentialScheduler>
</AdTracker>
```

3.9.3 Sample: Play Spark File on Timeout

Plays an image while waiting for motion. If no motion is detected within five seconds, a timeout will occur and the specified Spark file will play. If motion is detected within five seconds another image will play for five seconds before returning to the first image and waiting for motion again.

waitformotion2\root.spark

```
<AdTracker AdName="Manual_Samples\WaitForMotion2">
  <SequentialScheduler>
    <WaitForMotion
      PointsThreshold="200"
      TimeoutRule = "PlaySparkOnTimeout"
      Timeout = "5"
      TimeoutSparkFile=".\image.spark"
      >
        <Image Name=".\Images\earth.jpg" />
    </WaitForMotion>
    <Image Name=".\Images\moon.jpg" Duration="5"/>
  </SequentialScheduler>
</AdTracker>
```

4 Play Items - Effects

Effect play items provide visual and interactive effects.

The standard effects included are described in the following sections.

AlphaMask 45

AlphaOverlay 47

AlphaTransition 49

BasicTiles 51

ColourKeying 59

FishSwarm 62

FluidDynamics 65

GenericSwarm 67

Greyscale 74

IceCrack 76

ParticlesEmitMotion 82

ParticlesImageSwitch 89

ParticlesInteract 92

ParticlesInteractAnimate 97

ParticlesInteractVapor 111

ParticlesReset 115

Rotate 118

Silhouette 119

WaterFX 120

WaterMask 129

WaterOverlay 131

WipeRevealExpand 133

WipeRevealNormal 138

WipeRevealReset 141

4.1 AlphaMask

The **AlphaMask** effect creates a mask over a child play item by specifying an ARGB image. The transparency (Alpha) value in the image controls what and how much of the child item is visible.

4.1.1 Attributes

AlphaImage *(Required)*

Specifies a 32-bit ARGB file that acts as a mask for a child play item. The Alpha setting in the file controls how much of the child play item is visible. The Alpha channel cannot be controlled by Dazzler; Dazzler simply reads the Alpha setting from the file.

Value Specifications: Type: FilePath

If the image specified in the **AlphaImage** attribute has no Alpha channel or transparency information, it will display as a solid image.

Image file names must be comprised of letters, numbers, underscores and dashes only.

4.1.2 Sample: Simple AlphaMask

Creates an **AlphaMask** over a water effect. The **AlphaImage** allows the **WaterFX** play item to show through based on the alpha mask's transparency.

AlphaMask\root.spark

```
<AdTracker AdName="Manual_Samples\AlphaMask">
  <AlphaMask AlphaImage=".\Images\logo.tga">
    <WaterFX
      Density="0.96"
      Speed="20"
      RefractionFactor="1.0"
      Height="0.05"
      >
        <Image Name=".\Images\bg_blue.jpg"/>
    </WaterFX>
  </AlphaMask>
</AdTracker>
```

4.2 AlphaOverlay

This **Effect** play item uses a shader filter to apply alpha (transparency) to other play items.

4.2.1 Attributes

AlphaEnabled *(Default)*

When set to "True", alpha blending is enabled.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

g_fAlpha *(Default)*

Specifies the alpha blending factor that will be applied to affected child items.The lower the value, the greater the transparency.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

Name *(Required)*

Specifies **AlphaOverlay** as the shader filter type.

Value Specifications: Type: FilePath

To use **AlphaOverlay** as the shader filter type, it is necessary to specify "VisualFilters\AlphaOverlay.pso" as the **Name** attribute. See sample code for an example.

Notes

1. The **AlphaOverlay** shader filter must be the second item within a **ChildPlayer**. The first item in the **ChildPlayer** will be visible through the transparent regions of the second item (see sample code). For more information on the **ChildPlayer** play item, see ChildPlayer 33.
2. The **AlphaOverlay** shader filter must include one child play item, which is the item to which alpha is applied (see sample code).
3. Nesting shader filters is not supported. You cannot apply a shader filter to an item that already has a shader filter applied to it.

4.2.2 Sample: AlphaOverlay

Applies 50% alpha blending to an image superimposed over another image.

AlphaOverlay\root.spark

```
<AdTracker AdName="Manual_Samples\AlphaOverlay">
  <ChildPlayer>
    <Image Name=".\Images\earth.jpg" />
    <ShaderFilter
      Name="VisualFilters\AlphaOverlay.pso"
      AlphaEnabled="True"
      g_fAlpha="0.5"
      >
          <Image Name=".\Images\logo.jpg"/>
    </ShaderFilter>
  </ChildPlayer>
</AdTracker>
```

4.3 AlphaTransition

This is an **Effect** play item that provides an automatic fade transition from one child play item to another.

This **Effect** does not respond to motion, but to time, as set in its **DelayTime** attribute.

4.3.1 Attributes

DelayTime *(Default)*

Sets the time, in seconds, before fade begins.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 2.0

FadeTime *(Default)*

Controls how long, in seconds, the actual fade transition takes.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 3.0

Loop *(Default)*

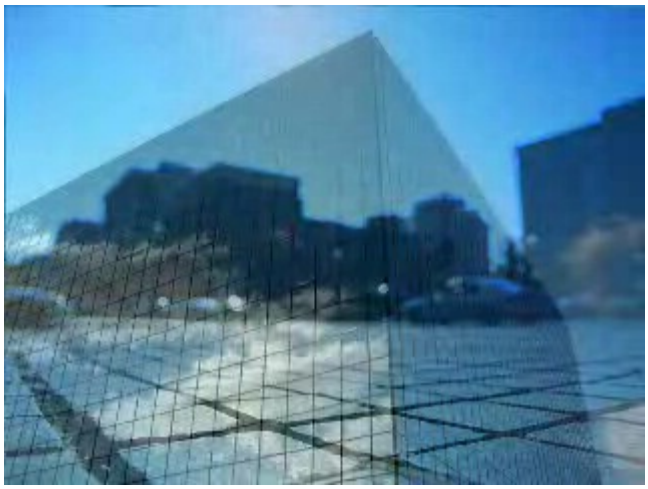
Sets the loop control for the entire **AlphaTransition** play item. When set to "True", the last child play item will fade to reveal the first item in the child play item list and continue through the other items as long as the **AlphaTransition** play item is active. If set to "False", the **AlphaTransition** play item will stop when the last child play item is revealed.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

Notes

- When this play item is within a SequentialScheduler 39 and the **Loop** attribute is set to "False", the play item will, upon its final transition, terminate, thus allowing the next play item within the SequentialScheduler 39 to start, regardless of the value of the Duration 20 attribute. For example, if an **AlphaTransition** play item inside a SequentialScheduler 39 has its Duration 20 value set to 30 seconds and its **Loop** value set to "False", but it only takes 12 seconds to display all of its child play items, then **AlphaTransition** will end after 12 seconds and the SequentialScheduler 39 will immediately move on to its next play item.

4.3.2 Sample: Simple AlphaTransition



Creates a fade transition between two **Image** play items that loop indefinitely. Each item remains on screen for three seconds. The fade transition also takes three seconds to complete.

AlphaTransition\root.spark

```
<AdTracker AdName="Manual_Samples\AlphaTransition">
  <AlphaTransition
    DelayTime="3.0"
    FadeTime="3.0"
    Loop="True"
    >
      <Image Name=".\Images\pavement.jpg"/>
      <Image Name=".\Images\tower.jpg"/>
  </AlphaTransition>
</AdTracker>
```

4.4 BasicTiles

This is a transition tile **Effect** that, upon motion detection flips from one child play item to another based on a specified tile layout. As the flip occurs, the successive item is visible on the back of the current item. The actual tiles that flip depend on the area of motion detection. Various flip effects can be produced depending on the values of the horizontal and vertical tile attributes, including checkerboard, Venetian blinds, twirling strips and the flipping of an entire item at once.

ScreenXtreme Users! See the special **BasicTiles** sample using the **KeyedCameraViewer**:

Venetian Blinds with Video 158

4.4.1 Attributes

AutoFlipTimeout (*Default*)

If no motion is detected for the time specified (in seconds), all remaining tiles will automatically flip.

Value Specifications: Type: Decimal; Range: 1.0 and up; Default: 10.0

BackgroundColor (*Default*)

An RGB value that sets the color that will appear behind the tiles as they flip. For more information about setting color attributes, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: "0.0, 0.0, 0.0"

CompletionSound (*Optional*)

Specifies the path to a sound file that plays when all tiles for an item are flipped.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

CompletionSoundVolume (*Dependent*)

Sets the volume for the file specified by the optional **CompletionSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

CompletionThreshold (*Default*)

Sets the percentage of tiles that needs to be flipped before all remaining tiles are automatically flipped. A setting of "1.0" (100%) essentially turns off this attribute. Note that the **AutoFlipTimeout** attribute, if present, takes precedence over this one.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

Direction (*Default*)

The direction in which the tiles flip. "X" (horizontal) causes the tiles to flip top to bottom. "Y" (vertical) cause the tiles to flip side to side. "B" (both) cause the tiles to flip alternating horizontally and vertically.

Value Specifications: Type: Enumerated; Options: "X", "Y", or "B"; Default: "X"

HorizontalTiles (*Default*)

Specifies the number of tiles across the presentation area.

Value Specifications: Type: Integer; Range: 1 - 64; Default: 5

The number of tiles is limited by your computer's resources.

Loop (*Default*)

Sets the loop control for the entire **BasicTiles** play item. When set to "True", the last child play item will flip to reveal the first item in the child play item list and continue through the other items as long as the **BasicTiles** play item is active. If set to "False", the **BasicTiles** play item will stop when the last child play item is revealed.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

MaxSounds (*Dependent*)

When the **TileFlipSound** attribute is used, sets the maximum number of concurrent sound effects allowed during tile flipping.

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 6

MinResetFlipTimeout, **MaxResetFlipTimeout** (*Optional*)

Sets an optional range of time, in seconds, when all flipped tiles will revert to the previously revealed image, but only if all tiles have not yet flipped to fully reveal next image. A random time will be chosen based on the specified range.

Value Specifications: Type: Decimal; Range: 0.0 and up

Perspective (*Default*)

Enables or disables perspective correction on the rotating tiles. When **Perspective** is set to "True", as the tiles flip, they look as if their edges get larger and then smaller as they flip over.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

RotationSpeed (*Default*)

How fast (in degrees per second) each tile flips.

Value Specifications: Type: Decimal; Range: 0.1 - 1440.0 ; Default: 45.0

TileFlipSound (*Optional*)

Specifies the path to a sound file that plays each time a set of tiles flip. Concurrent playing of the specified sound effect is supported and controlled by the **MaxSounds** attribute.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

TouchReset (*Default*)

If set to "True", and all tiles have not yet flipped to fully reveal next image, individual tiles will, upon interaction, flip back to reveal previous image.

When set to "True", and if the effect hasn't timed out or moved on to the next effect, you will be able to flip the tiles again.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

TransitionDelay (*Default*)

Sets the amount of time that interaction will be disabled once an item is completely flipped. This enables the user to see the new item for the time specified before starting the flip effect again.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 2.0

VerticalTiles (*Default*)

Specifies the number of tiles from top to bottom of the presentation area.

Value Specifications: Type: Integer; Range: 1 - 64; Default: 5

The number of tiles is limited by your computer's resources.

Volume (*Dependent*)

Sets the volume for the file specified by the optional **TileFlipSound** attribute.

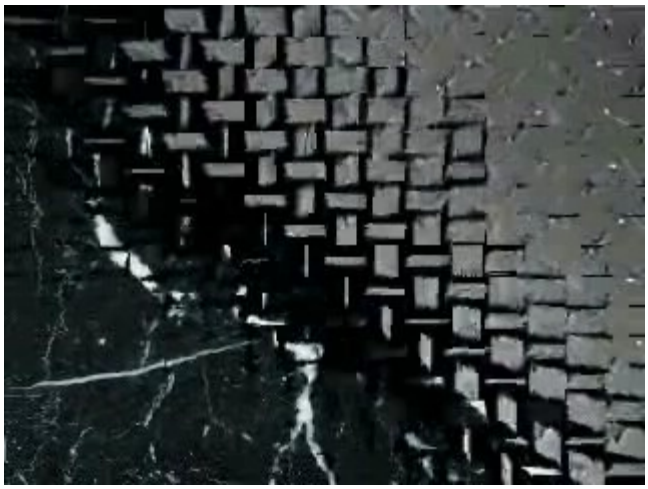
Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

Notes

- If a Duration 20 attribute is set, all remaining tiles automatically flip just before the duration period expires.
- In order to create a series of different tile effects, you would need to create a sequence of **BasicTiles** play items within a **Scheduler** play item. In order to ensure clean transitions between instances of **BasicTiles** play items, the last child play item in one **BasicTiles** play item should be the same as the first child play item in the subsequent one. The **Loop** attribute should be set to "False" in each instance.
- When this play item is within a SequentialScheduler 39 and the **Loop** attribute is set to "False", the play item will, upon its final transition, terminate, thus allowing the next play item within the SequentialScheduler 39 to start, regardless of the value of the Duration 20 attribute.

  For example, if a **BasicTiles** effect inside a SequentialScheduler 39 has its Duration 20 value set to 30 seconds and its **Loop** value set to "False", but it only takes 12 seconds to flip through and display all of its child play items, then the **BasicTiles** effect will end after 12 seconds, and the SequentialScheduler 39 will immediately move on to the next play item in the list.

4.4.2 Sample: BasicTiles Checkerboard

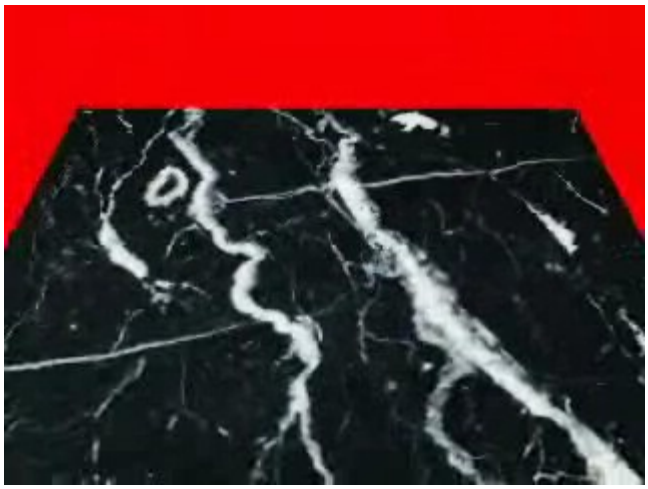


Creates a 16 x 16 (256 tiles in total) checkerboard tile grid effect that flips in both directions. The time of each tile flip is exactly one second. The background color is black. Perspective correction is enabled. As soon as 80% of the tiles are flipped the rest of the item will flip automatically. Regardless of motion detection each item will completely flip in 2 minutes (120 seconds). The play item will transition through two child play items. After flipping to a new item there will be a delay of 2 seconds before the play item responds to motion detection again. The last item will flip to the first item and continue indefinitely. Sound effects are at 40 per cent volume for both the flip action and when the current item is completed. The flip action sound effect allows for up to 6 concurrent sounds.

BasicTiles_CheckerBoard\root.spark

```
<AdTracker AdName="Manual_Samples\BasicTiles_Checkerboard">
  <BasicTiles
    HorizontalTiles="16"
    VerticalTiles="16"
    RotationSpeed="180.0"
    Direction="B"
    BackgroundColor="0.0,0.0,0.0"
    Perspective="True"
    CompletionThreshold="0.8"
    AutoFlipTimeout="120"
    TransitionDelay="2.0"
    Loop="True"
    TileFlipSound=".\Sounds\TileFlip.wav"
    Volume="0.4"
    MaxSounds="6"
    CompletionSoundEffect=".\Sounds\TileComplete.wav"
    CompletionSoundVolume="0.4"
    >
      <Image Name=".\Images\marble.jpg"/>
      <Image Name=".\Images\metal.jpg"/>
  </BasicTiles>
</AdTracker>
```

4.4.3 Sample: BasicTiles Entire Image Flip

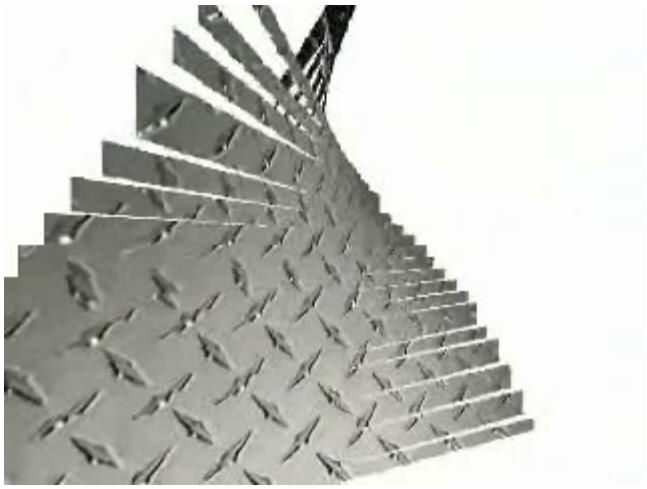


Flips entire item horizontally. The time of the flip is two seconds. The background color is red. Perspective correction is enabled. The play item will transition through two child play items. After flipping to the next item there will be a delay of 2 seconds before the play item responds to motion detection again. The last item will flip to the first item and continue indefinitely.

BasicTiles_ImageFlip\root.spark

```
<AdTracker AdName="Manual_Samples\BasicTiles_ImageFlip">
  <BasicTiles
    HorizontalTiles="1"
    VerticalTiles="1"
    RotationSpeed="90.0"
    Direction="X"
    BackgroundColor="1.0,0.0,0.0"
    Perspective="true"
    CompletionThreshold="0.8"
    TransitionDelay="2.0"
    Loop="True"
    >
      <Image Name=".\Images\marble.jpg"/>
      <Image Name=".\Images\metal.jpg"/>
  </BasicTiles>
</AdTracker>
```

4.4.4 Sample: BasicTiles Twirling Strips



Creates thin twirling horizontal strips. The time of each flip is two seconds. The background color is white. Perspective correction is enabled. As soon as 80% of the tiles are flipped the rest of the item will flip automatically. The play item will transition through two child play items. After flipping to a new item there will be a delay of two seconds before the play item responds to motion detection again. The last item will flip to the first item and continue indefinitely.

BasicTiles_TwirlingStrips\root.spark

```
<AdTracker AdName="Manual_Samples\BasicTiles_TwirlingStrips">
  <BasicTiles
    HorizontalTiles="1"
    VerticalTiles="30"
    RotationSpeed="90.0"
    Direction="Y"
    BackgroundColor="1.0,1.0,1.0"
    Perspective="True"
    CompletionThreshold="0.8"
    TransitionDelay="2.0"
    Loop="True"
    >
      <Image Name=".\Images\marble.jpg"/>
      <Image Name=".\Images\metal.jpg"/>
  </BasicTiles>
</AdTracker>
```

4.4.5 Sample: BasicTiles Venetian Blinds

Creates a vertical Venetian blind effect. The time of each tile flip is less than one second. The background color is grey. Perspective correction is enabled. As soon as 80% of the tiles are flipped, the rest of the item will flip automatically. The play item will transition through two child play items. After flipping to a new item there will be a delay of 2 seconds before the play item responds to motion detection again. The last item will flip to the first item and continue indefinitely.

BasicTiles_VenetianBlinds\root.spark

```
<AdTracker AdName="Manual_Samples\BasicTiles_VenetianBlinds">
  <BasicTiles
    HorizontalTiles="10"
    VerticalTiles="1"
    RotationSpeed="270.0"
    Direction="Y"
    BackgroundColor="0.5,0.5,0.5"
    Perspective="True"
    CompletionThreshold="0.8"
    TransitionDelay="2.0"
    Loop="True"
    >
      <Image Name=".\Images\marble.jpg"/>
      <Image Name=".\Images\metal.jpg"/>
  </BasicTiles>
</AdTracker>
```

4.5 ColourKeying

This is an **Effect** play item that uses a shader filter to remove a color or color range from a displayed image in order to make transparent regions.

4.5.1 Attributes

g_fClrTolerance *(Default)*

Sets the tolerance level of the specified key color, whereby "0.0" indicates no tolerance (an exact match is needed for a pixel to be rendered transparent, and "1.0" indicates absolute tolerance (whatever key color is specified, all pixels will be transparent).

Generally, a relatively small tolerance level provides best results to ensure that only those pixels in the desired color range are included. A value of "0.0" is not recommended, since in most cases, it is necessary to include minor variations in the key color.

Due to anti-aliasing (the process whereby the appearance of smoothness is produced by blurring the edges of an image), the edges of the opaque areas of a keyed image may appear to retain colors similar to the key color. This can be rectified by increasing the tolerance level. Be careful not to raise the tolerance level too high as it may cause unwanted transparent regions.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.009

g_vKeyClr (*Default*)

An RGB value that specifies the color to be keyed (made transparent). For more information about setting color attribute values, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.0,0.0,0.0 (black)

Name (*Required*)

Specifies **ColourKeying** as the shader filter type.

Value Specifications: Type: FilePath

To use **ColourKeying** as the shader filter type, it is necessary to specify "VisualFilters\ColourKeying.pso" as the **Name** attribute. See sample code for examples.

Notes

1. The **ColourKeying** shader filter must be the second item within a **ChildPlayer**. The first item in the **ChildPlayer** will be visible through the transparent regions of the second item (see sample code). For more information on the **ChildPlayer** play item, see ChildPlayer 33.
2. The **ColourKeying** shader filter must include one child play item, the item to which transparency is applied.
3. For best results, the key color (the color that will be transparent) should not be used elsewhere in the affected image since those areas will also be made transparent.
4. Nesting shader filters is not supported. You cannot apply a shader filter to an item that already has a shader filter applied to it.

4.5.2 Sample: Keyed WaterFX Over an Image

Creates a transparent water effect over an image.

ColourKeying1\root.spark

```
<AdTracker AdName="Manual_Samples\ColourKeying1">
  <ChildPlayer>
    <Image Name=".\Images\earth.jpg" />
    <ShaderFilter
      Name="VisualFilters\ColourKeying.pso"
      g_vKeyClr="1.0,1.0,1.0"
      g_fClrTolerance="0.43"
      >
        <WaterFX
        Density="0.98"
        Speed="20.0"
        RefractionFactor="1.0"
        Height="0.05"
        InteractionMethod="Solid"
        >
          <Image Name=".\Images\logo.jpg"/>
        </WaterFX>
    </ShaderFilter>
  </ChildPlayer>
</AdTracker>
```

4.5.3 Sample: Keyed Flash File Over Thick Gel

Creates a keyed Flash file over a thick gel effect. The Flash file is an interactive donut on a white background, which has been rendered transparent. A tolerance value of "0.5" is used to remove the edges of the donut. The donut is layered over a **WaterFX** play item that is set to look like thick gel over a chocolate-like background.

ColourKeying2\root.spark

```
<AdTracker AdName="Manual_Samples\ColourKeying2">
  <ChildPlayer>
    <WaterFX
      Density="0.99"
      Speed="0.1"
      RefractionFactor="1.0"
      Height="0.005"
      InteractionMethod="Solid"
      >
        <Image Name=".\Images\choco.jpg" />
    </WaterFX>
    <ShaderFilter
      Name="VisualFilters\ColourKeying.pso"
      g_vKeyClr="1.0,1.0,1.0"
      g_fClrTolerance="0.5"
      >
        <FlashFile Name=".\Swf\donut.swf" />
    </ShaderFilter>
  </ChildPlayer>
</AdTracker>
```

4.6 FishSwarm

This is a swarm **Effect** play item in which multiple instances of an animated sequence move about the presentation area. The number of entities and various aspects of their behavior may be set.

4.6.1 Attributes

FishCount *(Default)*

Specifies the number of entities that will appear.

Value Specifications: Type: Integer; Range: 1 - 256; Default: 20

The number of entities is dependant on system resources.

FleeSpeed *(Default)*

Specifies the speed, in pixels per second, at which entities will move in response to interaction.

For best results **FleeSpeed** should be equal to or greater than **MaxRoamSpeed**.

Setting a very high **FleeSpeed** (more than 4500), may return unpredictable results.

Value Specifications: Type: Integer; Range: 1 -10000; Default: 700

MinRoamSpeed; **MaxRoamSpeed** *(Default)*

Creates a range, in pixels per second, for the normal roaming speed of the entities. Except when in its flee state, each entity will move about the screen at a speed based on a random value between the set minimum and maximum. Each entity moves about at its own assigned speed.

MaxRoamSpeed must be equal to or greater than **MinRoamSpeed**.

Value Specifications: Type: Integer; 0 and up; Default: ***MinRoamSpeed****: 100;* ***MaxRoamSpeed****: 500*

ScreenRadius *(Dependent))*

If **UseCircularScreen** is set to "True", specifies the radius, in pixels, of the circular area in which the entities can move.

Creating a radius that crowds the entities is not recommended.

Value Specifications: Type: Integer; Range: 100 and up; Default: 400

StartFleeDistance *(Default)*

The maximum distance (in pixels) from the user interaction, that an entity can be before starting to move away.

Value Specifications: Type: Integer; Range: 1 -1000; Default: 50

SwimAnimationPath *(Required)*

Specifies the path to the folder which contains the image files which make up the animation sequence for the entities. The animation sequence will repeat continuously.

Value Specifications: Type: FilePath

Notes

1. Images must be 32-bit, ARGB image files and have identical dimensions.

2. Image file names must be comprised of letters, numbers, underscores and dashes only.

3. Images should be oriented vertically with the head of the fish facing upwards as follows:

4. Transparency is set by an image's alpha channel.

5. The order of the animation sequence is based on the file names of the images according to standard Windows® alphanumeric sorting conventions.

Example

Files would be sorted as follows:

file01.img

file10.img

file11.img

file2.img

file4.img

TurningSharpness *(Default)*

Specifies how much, in degrees per second, the entity will turn when reacting to user interaction or when meeting a boundary. Turning direction is random.

Value Specifications: Type: Decimal; Range: 1.0 - 1440.0; Default: 180.0

UseCircularScreen *(Default)*

When set to "True", allows for the setting of a circular area that limits where the entities can move. See **ScreenRadius** to set the area size.

Value Specifications: Type: Boolean; Range: "True" of "False"; Default: "False"

UseCircularScreen is best used by making **FishSwarm** a child element of AlphaMask 45.

4.6.2 Sample: FishSwarm

Creates a **FishSwarm** effect over an **Image** play item. Twenty entities are spawned. They move at a random speed between 50 and 150 pixels per second. They will turn away at a 180 degree angle at, a speed of 350 pixels per second, when they are at a distance of 50 pixels or less from the user interaction. They will return to their normal state when they are 500 pixels away from user interaction.

FishSwarm\root.spark

```
<AdTracker AdName="Manual_Samples\FishSwarm">
  <FishSwarm
    FishCount="20"
    StartFleeDistance="50"
    EndFleeDistance="500"
    MinRoamSpeed="50.0"
    MaxRoamSpeed="150.0"
    TurningSharpness="180"
    FleeSpeed="350.0"
    SwimAnimationPath=".\Images\fish_swim_128"
    >
      <Image Name=".\Images\bg_blue.jpg"/>
  </FishSwarm>
</AdTracker>
```

4.7 FluidDynamics

This is a Fluid dynamics **Effect** play item. Fluid effect forms as user interacts with presentation area. Effect behavior, including formation and dissipation are based on attribute settings.

The colors of the fluid effect are based on a 256-color palette. As the intensity of the effect grows, higher color values are displayed. Several sample palettes are included, but custom palettes may be used as well.

This effect provides good results with or without a child play item.

4.7.1 Attributes

AlphaBlend *(Default)*

When set to "True" alpha blending is activated. Blending is relative to density.

Value Specifications: Type: Boolean; Range: "True" of "False"; Default: "False"

BlurFactor *(Default)*

Amount of blur to apply to the effect. The higher the value, the greater the amount of blur applied.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.5

Density *(Default)*

Specifies the amounts of density to insert along the tracker outline points.The higher the value, the greater the intensity of the fluid effect.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 100.0

DensityDissipation *(Default)*

Sets the density dissipation. The higher the value, the slower the density dissipates.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.1

Setting **DensityDissipation** to "0.0" effectively turns off the effect.

Force *(Default)*

Sets the amount of force applied upon user interaction. Force is inserted from the middle of the tracker outline points and direction is relative to the center of the screen. The higher the value, the greater amount of fluid movement.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 1000.0

ForceDissipation *(Default)*

Sets the force dissipation. The higher the value, the slower the force dissipates.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.8

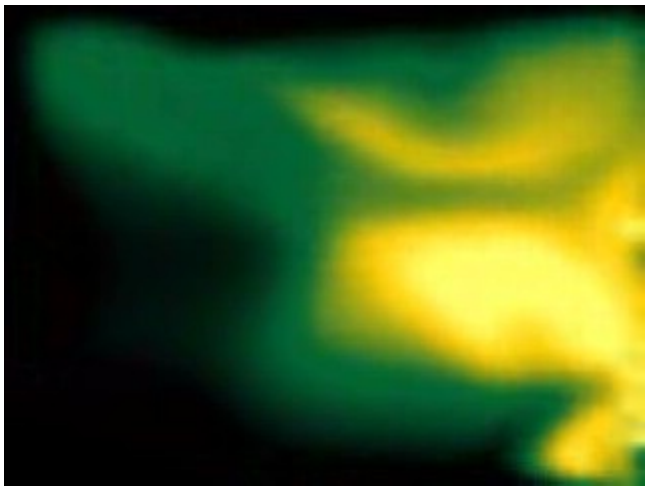
Palette *(Default)*

Specifies the file path to the 256-color palette file. As density increases, the color value displayed increases.

If a palette is not specified, a default palette will be used.

Value Specifications: Type: FilePath

4.7.2 Sample: FluidDynamics



A simple **FluidDynamics** effect, using the default color palette. No child play item is being used.

FluidDynamics\root.spark

```
<AdTracker AdName="Manual_Samples\FluidDynamics">
  <FluidDynamics
    Force="2000.0"
    Density="150.0"
    ForceDissipation="0.1"
    DensityDissipation="0.1"
    BlurFactor="0.5"
    AlphaBlend="True"
    >
  </FluidDynamics>
</AdTracker>
```

4.8 GenericSwarm

This is a swarm **Effect** play item that allows interactive animated entities to move about in various states as described in the Attributes 67 section. The number of entities and various aspects of their behavior may be set.

For best results it is recommended that the Tracker be set to the Background Model tracking mode (GestureFX only) when using this play item. See the GestureFXTracker documentation for more information. Not recommended for ScreenXtreme.

4.8.1 Attributes

AnimationSpeed *(Default)*

Specifies, in frames per second, the animation sequence speed of the entities.

This is not the speed that the animations move on screen, but rather the speed of the various entity state animation sequences. The speed at which the entities move on screen is set by the **RoamSpeed**, **AttractSpeed**, and **FleeSpeed** attributes.

Value Specifications: Type: Integer; Range: 1 - 10000; Default: 30

AttractAnimationPath (SwimAnimationPath), **ConvertAnimationPath**, **FleeAnimationPath** *(Required)*

These three required attributes specify the paths to the folders containing the image files which make up the animation sequences for each of the three different states the entities can be in. The three states (along with their corresponding attributes) are defined as follows:

Attract state (AttractAnimationPath/SwimAnimationPath)

This is the state the entities are in when they are simply roaming about and when they are attracted to the tracker outline. If an entity does not detect the tracker outline, is being attracted to the tracker outline, or is attached to the tracker outline, it is this animation sequence that plays. Entities will use this animation sequence continually until they enter the **Convert state**.

Convert state (ConvertAnimationPath)

When an entity is attached to the tracker outline silhouette for the amount of time specified in the **FleeTime** attribute, this animation sequence begins. The **Convert state** animation sequence will play only once and is immediately followed by the **Flee state** animation sequence.

Flee state (FleeAnimationPath)

If and when an entity plays through the **Convert state** animation sequence, it will begin to play the **Flee state** animation sequence. The **Flee state** animation sequence will play continually until the entity moves completely off screen.

Value Specifications: Type: FilePath

Notes

1. Images must be 32-bit, ARGB image files and images for all states must have identical dimensions.

2. Image file names must be comprised of letters, numbers, underscores and dashes only.

3. Images should be oriented vertically with the head of the entity facing upwards as follows:

4. Transparency is set by an image's alpha channel.

5. The order of the animation sequences is based on the file names of the images according to standard Windows® alphanumeric sorting conventions.

Example

Files would be sorted as follows:

file01.img

file10.img

file11.img

file2.img

file4.img

AttractSpeed *(Default)*

Sets the speed, in pixels per second, at which the entities will move when they are attracted to the tracker outline.

While the **AttractAnimationPath** attribute specifies the animation for an entity both when it is roaming or being attracted to the tracker outline, **AttractSpeed** applies only when it is attracted to the tracker outline.

Value Specifications: Type: Integer; Range: 0 - 10000; Default: 150.0

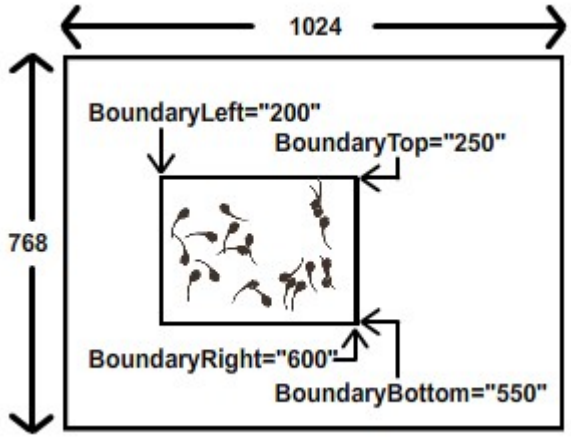
BoundaryLeft, **BoundaryRight**, **BoundaryTop**, **BoundaryBottom** *(Optional)*

These attributes specify, in pixels, a rectangular screen boundary. The entities will only roam and interact within the boundary area. The rectangular boundary may be placed anywhere within the screen. **BoundaryLeft** and **BoundaryRight** correspond to the left and right sides of the rectangle respectively, measured from the left side of the screen area. **BoundaryTop** and **BoundaryBottom** correspond to the top and bottom sides of the rectangle respectively, measured from the top of the screen area.

Sample

The following illustration demonstrates a rectangular screen boundary of 400 pixels wide and 300 pixels high on a full screen resolution of 1024 X 768 using these attribute values:

BoundaryLeft="200", BoundaryRight="600", BoundaryTop="250", BoundaryBottom="550"



Notes

1. When specifying a rectangular screen boundary entities will spawn outside the rectangular boundary, but will only begin to roam and interact within the rectangular boundary.

2. **UseCircularScreen** takes precedence over these rectangular boundary settings. When **UseCircularScreen** is set to "True", **BoundaryLeft**, **BoundaryRight**, **BoundaryTop**, and **BoundaryBottom** will be ignored.

3. When specifying either **BoundaryLeft** or **BoundaryTop**, it is necessary to also specify the corresponding **BoundaryRight** or **BoundaryBottom** even when wanting to include the far right or bottom of the screen. Failure to include the corresponding attributes will have undesired results.

4. **BoundaryRight** must be greater than **BoundaryLeft**, and **BoundaryBottom** must be greater than **BoundaryTop**.

Value Specifications: Type: Integer; Range: 0 - screen size

EntityCount *(Default)*

Specifies the number of entities that will appear.

The number of entities depend on system resources.

When **UseCircularScreen** is set to "False", there is a tendency for all the entities to eventually move off screen without being replaced by new ones. Therefore when wanting the entities to swarm full screen, it is recommended to set **UseCircularScreen** to "True" and specify a radius that will encompass the whole screen. See the GenericSwarm1.spark 72 sample for an example.

Value Specifications: Type: Integer; Range: 1 - 256; Default: 20

FleeSpeed *(Default)*

Sets the speed, in pixels per second, in which the entities will move as soon as they enter their **Convert state** and continuing into their **Flee state**.

Value Specifications: Type: Integer; Range: 1 - 10000; Default: 150.0

FleeTime *(Default)*

Sets the time, in seconds, that an entity will need to be attached to the tracker outline before it enters its **Convert state**.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 5.0

MinAttractionDistance, **MaxAttractionDistance** *(Default)*

An entity will begin to move toward the tracker outline when it is at a random distance to the tracker outline within the range set by these two values. Values are measured in pixels.

MaxAttractionDistance must be equal to or great than **MinAttractionDistance**.

Value Specifications: Type: Integer; Range: 1 - 1000; Default: **MinAttractionDistance**: 25, **MaxAttractionDistance**: 500

RoamSpeed *(Default)*

Sets the speed, in pixels per second, in which the entities will move when they are roaming about the screen area. The entities are in their "roam" state whenever they are not being attracted to the tracker outline, not in their **Convert state**, or not in their **Flee state**.

While the **AttractAnimationPath** attribute specifies the animation for an entity both when it is roaming or being attracted to the tracker outline, **RoamSpeed** applies only when it is roaming.

Value Specifications: Type: Integer; Range: 0 - 10000; Default: 50.0

ScreenRadius *(Dependent)*

Specifies the size, in pixels, of the screen radius when **UseCircularScreen** is set to "True". For more information, see **UseCircularScreen**.

Value Specifications: Type: Integer; Range: 100.0 - 10000.0; Default: 384.0

UseCircularScreen *(Default)*

When set to "True", the entities will only roam and interact within the radius specified in the **ScreenRadius** attribute. The specified radius is measured from the center of the screen. To create a rectangular boundary area that can be placed anywhere on the screen, see the **BoundaryLeft**, **BoundaryRight**, **BoundaryTop**, and **BoundaryBottom** attributes.

Notes

1. When **UseCircularScreen** is set to "True", entities will spawn outside of the specified radius, but will only begin to roam and interact within the specified radius.

2. **UseCircularScreen** takes precedence over the **BoundaryLeft**, **BoundaryRight**, **BoundaryTop**, and **BoundaryBottom** attributes. When **UseCircularScreen** is set to "True", the **BoundaryLeft**, **BoundaryRight**, **BoundaryTop**, and **BoundaryBottom** attribute settings will be ignored.

3. When **UseCircularScreen** is set to "False", there is a tendency for all the entities to eventually move off screen without being replaced by new ones. Therefore when wanting the entities to swarm full screen, it is recommended to set **UseCircularScreen** to "True" and specify a radius that will encompass the whole screen. See the GenericSwarm1.spark 72 sample for an example.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

4.8.2 Sample: GenericSwarm

A simple **GenericSwarm** sample using default values except for the number of entities and their flee time. Thirty tadpoles will be attracted to the tracker outline when they are between 25 and 500 pixels away, upon which they will convert to frogs after 1 second and flee away. To ensure that the tadpole entities will continue to spawn, a screen radius of 640 pixels has been specified. This is the radius required to encompass a screen resolution of 1024 X 768.

To see this sample, but with the tracker outline silhouette visible, see GenericSwarm with Silhouette 73.

GenericSwarm1\root.spark

```
<AdTracker AdName="Manual_Samples\GenericSwarm1">
  <GenericSwarm
    AttractAnimationPath=".\Images\attract"
    ConvertAnimationPath=".\Images\convert"
    FleeAnimationPath=".\Images\flee"
    EntityCount="30"
    AnimationSpeed="30"
    MinAttractionDistance="25"
    MaxAttractionDistance="500"
    AttractSpeed="150.0"
    FleeTime="1.0"
    FleeSpeed="150.0"
    UseCircularScreen="True"
    ScreenRadius="640"
    >
      <Image Name=".\Images\dazzler1024.jpg"/>
  </GenericSwarm>
</AdTracker>
```

4.8.3 Sample: GenericSwarm with Silhouette

Similar to GenericSwarm Sample 72, except it has been made the child item of a **Silhouette** play item so that the tracker outline silhouette can be seen.

GenericSwarm2\root.spark

```
<AdTracker AdName="Manual_Samples\GenericSwarm2">
  <Silhouette
    Color = "1.0,0.0,0.0,0.6"
    BlurFactor = "0.6"
    >
      <GenericSwarm
        AttractAnimationPath=".\Images\attract"
        ConvertAnimationPath=".\Images\convert"
        FleeAnimationPath=".\Images\flee"
        EntityCount="30"
        AnimationSpeed="30"
        MinAttractionDistance="25"
        MaxAttractionDistance="500"
        AttractSpeed="150.0"
        FleeTime="1.0"
        FleeSpeed="150.0"
        UseCircularScreen="True"
        ScreenRadius="640"
        >
          <Image Name=".\Images\dazzlerFX1024.jpg"/>
      </GenericSwarm>
  </Silhouette>
</AdTracker>
```

4.9 Greyscale

This **Effect** play item applies a greyscale shader filter to the screen display. The filter may be applied to an entire play item (simple or complex) or to a child play item within a play item.

When using the **Greyscale** shader filter within a SequentialScheduler 39, remember to place the Duration 20 attribute within the filter rather than in the child item of the filter.

4.9.1 Attributes

g_vLuminance *(Default)*

Applies a luminance setting for the RGB color channels of each pixel displayed.

Value Specifications: Type: Decimal; Range: 0.00 - 1.00; Default: 0.33

For most full-color images, the default setting of "0.33,0.33,0.33" provides best results.

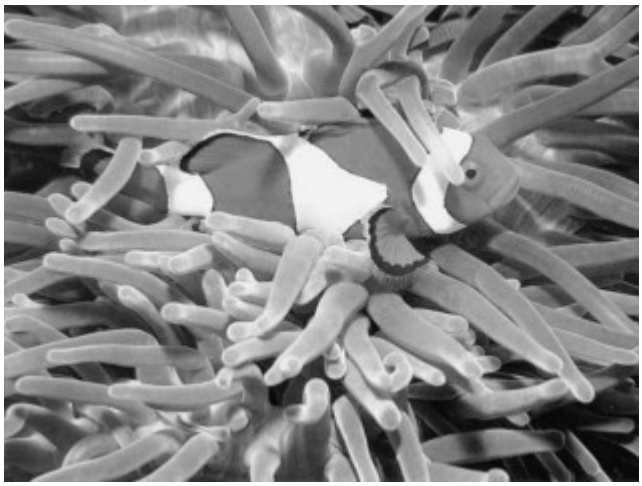
Name *(Required)*

Specifies **Greyscale** as the shader filter type. See sample code for an example.

Value Specifications: Type: FilePath

To use **Greyscale** as the shader filter type, it is necessary to specify "VisualFilters\Greyscale.pso" as the **Name** attribute.

4.9.2 Sample: Greyscale Shader Filter



Applies the **Greyscale** shader filter to an image using 33% luminance setting.

Greyscale\root.spark

```
<AdTracker AdName="Manual_Samples\Greyscale">
  <ShaderFilter
    Name="VisualFilters\Greyscale.pso"
    g_vLuminance="0.33,0.33,0.33"
    >
      <Image Name=".\Images\clown_fish.jpg"/>
  </ShaderFilter>
</AdTracker>
```

4.10 IceCrack

Upon motion detection, this **Effect** play item applies an ice crack effect to a primary child play item, so that the item will appear to look as if it is breaking up to reveal the secondary play item. Once the secondary item is completely revealed, if there is another play item listed, the secondary item will become the new primary item and the effect will begin again, and so on.

When cracked pieces are created, they will shrink in size upon additional motion detection until, according to a set threshold percentage, it will shrink and fade away.

All cracked pieces are three- or four-sided, randomly generated shapes. The number of shapes, their size, the variations of shape and how they work together are controlled by the play item attributes.

ScreenXtreme Users! See the special **IceCrack** sample using the **KeyedCameraViewer**:

IceCrack with Sound and Video 155

4.10.1 Attributes

ComplexityLevel *(Default)*

Determines the total number of cracked pieces that will be generated over the entire presentation area as follows:

Complexity Level	Number of pieces
6	64
7	128
8	256
9	512

Value Specifications: Type: Integer; Range: 6 - 9; Default: 8

CrackForce *(Default)*

Specifies the amount of force exerted upon motion detection. The greater the value, the more cracking will occur and the quicker the cracked pieces will shrink.

Value Specifications: Type: Decimal; Range: 0.1 - 1000.0; Default: 2.0

CrackRadius *(Default)*

Sets how much of the presentation area is affected by each instance of motion detection.

Value Specifications: Type: Integer; Range: 50 - 1000; Default: 128

SoundPath *(Optional)*

Specifies path to a folder containing one or more sound files that will play (randomly if there is more than one sound file) as cracking occurs. The timing of the files is controlled by the **MinSoundDelay** and **MaxSoundDelay** attributes. Files must be Windows®-compliant WAV files.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

FadeSpeed *(Default)*

Sets how long, in seconds, it takes a cracked piece to fade out once the **FadeThreshold** value is reached.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 1.0

FadeThreshold (Default)

Sets the percentage at which a cracked piece will no longer be affected by additional motion detection and will begin to shrink and fade away. The higher the value, the sooner the pieces will shrink and fade. Low values mean that pieces need to be relatively small before they begin to shrink and fade.

This setting is relative to each piece created. Therefore when using high complexity levels, which causes some pieces to be quite small, a low **FadeThreshold** would mean that these pieces would need be that much smaller in order to shrink and fade away.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.5

Loop *(Default)*

Sets the loop control for the entire **IceCrack** play item. When set to "True", the last item will reveal the first item in the child play item list and continue through the other items as long as the **IceCrack** play item is active. If set to "False", the **IceCrack** play item will stop once the last child play item is completely revealed.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

MaxSound *(Dependent)*

If **SoundPath** is specified, controls the maximum number of sounds that will play simultaneously.

Value Specifications: Type: Integer: Range: 1 - 64; Default: 12

MinSoundDelay, **MaxSoundDelay** *(Dependent)*

If **SoundPath** is specified, these two attributes control the delay, in seconds, between the playing of sound effects. Setting the same values for both minimum and maximum will force the sounds to play at regular intervals. Setting different values will create a random delay.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: ***MinSoundDelay****: 0.05,* ***MaxSoundDelay****: 2.0*

ShrinkSpeed *(Default)*

Sets how long, in seconds, it takes a cracked piece to shrink once the **FadeThreshold** value is reached.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 1.0

SplitMin, **SplitMax** *(Default)*

These two attributes control the shape variation of the cracked pieces. A value of 0.5 creates rectangular shapes. The more the values are less than or greater than 0.5, the more skewed the shapes will be. The wider the minimum/maximum range, the greater the variety of shapes. The more narrow the minimum/maximum range, the more uniform the shapes will be.

Setting both **SplitMin** and **SplitMax** to either "0.0" or "1.0" results in no cracking.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: SplitMin: 0.3, SplitMax: 0.4

TopLeft, **TopRight**, **BottomRight**, **BottomLeft** *(Default)*

Allows you to set the dimensions of the primary child play item which is also the "crackable" area. This has no effect upon the secondary child play item until it becomes the new primary child play item. Each of these attributes requires a pair of horizontal and vertical settings (for more information, see the **IceCrack** sample).

In most cases the default values produce the best results.

Values must be no bigger than your presentation area resolution (i.e. screen width and height as set in the Dazzler configuration [dazzler.cfg]).

Value Specifications: Type: Integer; Range: based on minimum and maximum screen resolution dimensions; Default: based on current screen resolution settings in Dazzler configuration (dazzler.cfg).

Volume *(Dependent)*

Sets the volume for the file specified by the optional **SoundPath** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

Notes

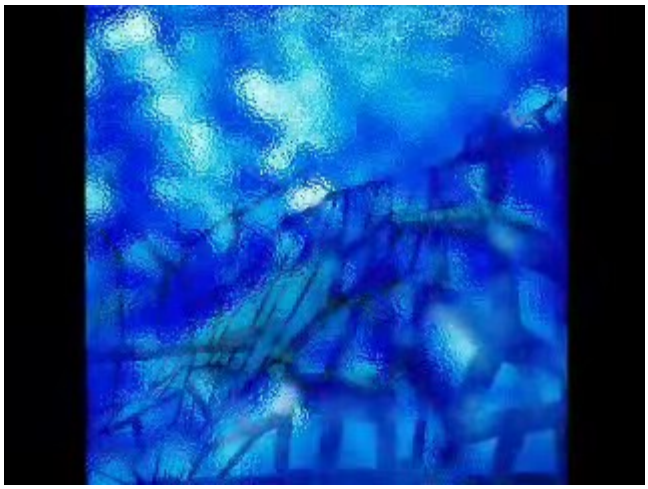
1. **MaxSoundDelay** must be equal to or greater than **MinSoundDelay**.

2. A wide range of delay time is more effective in situations where there is a lot of motion detection. Otherwise a narrow range of short delays is better.

3. When this play item is within a SequentialScheduler 39 and the Loop attribute is set to "False", the play item will, upon its final transition, terminate, thus allowing the next play items within the SequentialScheduler 39 to start, regardless of the value of the Duration 20 attribute.

For example, if an **IceCrack** play item inside a SequentialScheduler 39 has its Duration 20 value set to 30 seconds and its **Loop** value set to "False", but it only takes 12 seconds to display all of its child play items, then **IceCrack** will end, and the SequentialScheduler 39 will immediately move on to the next play item in the list.

4.10.2 Sample: IceCrack-Special Dimensions

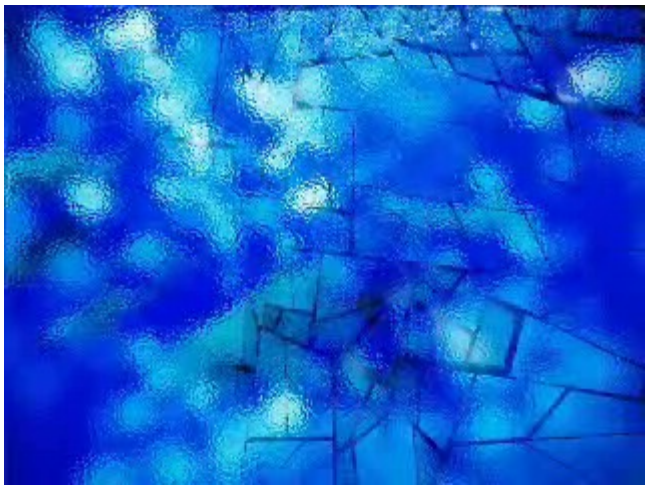


Creates a ice crack effect between two non-standard sized **Image** play items.

IceCrack2\root.spark

```
<AdTracker AdName="Manual_Samples\IceCrack1">
  <IceCrack
    ComplexityLevel="7"
    TopLeft="128,0"
    TopRight="896,0"
    BottomRight="896,768"
    BottomLeft="128,768"
    SplitMin="0.2"
    SplitMax="0.8"
    CrackForce="4.0"
    CrackRadius="128"
    FadeThreshold="0.8"
    ShrinkSpeed="2.0"
    FadeSpeed="5.0"
    Loop="True"
    >
      <Image Name=".\Images\icetop_cut.jpg"/>
      <Image Name=".\Images\iceshark_cut.jpg"/>
  </IceCrack>
</AdTracker>
```

4.10.3 Sample: IceCrack with Sound



Creates a normal ice crack effect between two **Image** play items.

IceCrack1\root.spark

```
<AdTracker AdName="Manual_Samples\IceCrack2">
  <IceCrack
    ComplexityLevel="7"
    SplitMin="0.2"
    SplitMax="0.8"
    CrackForce="2.0"
    CrackRadius="128"
    FadeThreshold="0.8"
    ShrinkSpeed="5.0"
    FadeSpeed="2.0"
    Loop="True"
    SoundPath=".\Sounds\ice"
    Volume="0.5"
    MaxSounds="12"
    MinSoundDelay="0.05"
    MaxSoundDelay="0.1"
    >
      <Image Name=".\Images\icetop.jpg"/>
      <Image Name=".\Images\iceshark.jpg"/>
  </IceCrack>
</AdTracker>
```

4.11 ParticlesEmitMotion

This **Effect** play item emits particles over the presentation area upon motion detection. Attributes control such things as the amount and appearance of particles as well as if and how they move.

The particle image files cannot exceed 1024x1024 in size and they must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

Suggested uses

ParticlesEmitMotion can be used to create effects such as bubbles, sparks, fire, smoke, waterfalls, etc.

While different kinds of motion effects can be created, all particles will move the same way based on the values specified in the **Velocity** and **Acceleration** attributes. In order to get the particles to move in different directions, you would need to nest at least one **ParticlesEmitMotion** play item inside another.

ScreenXtreme Users! See the special **ParticlesEmitMotion** sample using the **KeyedCameraViewer**:

Bubbles in Water with Video 154

4.11.1 Attributes

Acceleration *(Default)*

Sets the rate of increase of movement, in pixels per second, applied to the velocity of all particles upon spawning. Two values are available. The first value controls horizontal direction (a positive value denotes direction to the left; a negative value denotes direction to the right). The second value controls vertical direction (a positive value denotes downward direction; a negative value denotes upward direction). The default setting of "0.0, 0.0" denotes no acceleration.

Specifying acceleration values opposite to the velocity values will cause the particles to change direction from those specified in the **Velocity** settings.

Value Specifications: Type: Decimal; Range: -2000.0 - 2000.0; Default: "0.0, 0.0"

AnimationCycleTime *(Default)*

Time, in seconds, it takes to complete one animation sequence. The higher the number, the slower the sequence.

Value Specifications: Type: Decimal; Range: Greater than 0.0; Default: 1.0

AnimationMode *(Default)*

Controls animation behavior as follows:

1. **LOOP**: After playing the last image in the sequence, the animation will return to its first image and continue indefinitely.

2. **PINGPONG**: The animation sequence will continuously alternate between playing its images forward and reverse.

3. **ONESHOT**: The animation sequence will play through its images only once and stop at the last image. This also applies when **RandomStartFrame** is set to "True". In that case whatever frame the sequence begins with, the animation will still stop with the last one in the folder. If the random image happens to be the last one in folder, that particle will not animate at all.

4. **ONESHOTDIE**: Similar to ONESHOT, except that the particle disappears after the last frame is played.

Value Specifications: Type: Enumerated; Options: "LOOP", "PINGPONG", "ONESHOT", or "ONESHOTDIE"; Default: "LOOP"

BlendMode *(Default)*

Controls the blending of overlapping particles . "Normal" prevents color bleeding and saturation.

Value Specifications: Type: Enumerated; Options: "Normal" or "Saturate"; Default: "Normal"

EmitRatePerSecond *(Default)*

Specifies the number of particles spawned per second upon motion detection. The higher the value, the more particles will be spawned.

Value Specifications: Type: Integer; Range: 1 - 16383; Default: 350

The **EmitRate** (also now called **EmitRatePerFrame**) attribute is still supported. It spawns particles per frame instead of per second. Using **EmitRatePerSecond** provides more control and is recommended.

Very high emit rates may significantly slow down system performance. For best results, choose a value that gives you the look you want.

ExpandRate *(Default)*

Sets the time in seconds it takes for a particle to expand from point of appearing to its full size (the random size based on the **MinWidth**; **MaxWidth**; **MinHeight**; **MaxHeight** attributes). The smaller the value, the quicker the file will expand. Setting the expand time to "0.0" causes the particle to appear instantly at its size with no expansion effect.

The live time for a particle begins as soon as it appears, which begins from the start of expansion. If the expansion time is greater than the live time and a **FadeTime** greater than 0 is specified, then the particle will continue to expand while it is fading out.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 0.1

FadeTime *(Default)*

Sets the amount of time, in seconds, a particle will fade out once its live time has expired. The smaller the value, the quicker the particle will fade out. Setting the value to "0.0" will cause the particle to disappear instantly as soon as its live time has expired.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 0.5

MaintainAspectRatio *(Default)*

When set to "True" the height of a particular particle will always be the same as the random width based on the values specified in **MinWidth** and **MaxWidth**. **MinHeight** and **MaxHeight** values are ignored.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

MaxSoundSamples *(Dependent)*

If **SoundEffectPath** is specified, controls the maximum number of sounds that will play simultaneously.

Value Specifications: Type: Integer: Range: 1 - 1024; Default: 12

MinLiveTime, **MaxLiveTime** *(Default)*

Specifies the minimum and maximum amount of time, in seconds, that a particle will appear before beginning to fade out. The effect chooses a random time between the range set. **MaxLiveTime** must be equal to or greater than **MinLiveTime**. If the minimum and maximum times are identical, all particles with appear for the exact same amount of time before beginning to fade out.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 3.0

MinRotationSpeed, **MaxRotationSpeed** *(Default)*

Creates a range based on specified minimum and maximum values from which a random rotation speed, in degrees per second, will be applied to each particle. Each particle will be assigned its own random rotation speed. Positive values indicate counterclockwise direction; negative values indicate clockwise direction. "0.0" indicates no rotation.

MaxRotationSpeed must be equal to or greater than **MinRotationSpeed**.

Value Specifications: Type: Decimal; Range: -1440.0 - 1440.0; Default: "0.0"

MinSoundDelay, MaxSoundDelay *(Dependent)*

If **SoundEffectPath** is specified, these two attributes control the delay, in seconds, between the playing of sound effects. Setting different values will create a random delay between the playing of each sound. The minimum value must be equal to or less than the maximum value. Setting the same values for both minimum and maximum will force the sounds to play at regular intervals.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: **MinSoundDelay**: 0.05, **MaxSoundDelay**: 2.0

MinWidth; **MaxWidth**; **MinHeight**; **MaxHeight** *(Default)*

When a particle is spawned, it will appear and grow to a random pixel size within these four attributes. The maximums must be equal to or greater than the minimums. Setting identical minimum and maximum values will cause all particles to be the same size. Using various values creates a wide variety of particle sizes. If a minimum value is set greater than a maximum value, then the minimum value will be adjusted to be the same as the maximum value.

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 16

ParticleTexturePath *(Required)*

Specifies the path to the texture image file(s) upon which the particle images are based. Path may be to one file, to a folder of assorted images, a folder of subfolders, or a combination of assorted images and subfolders. If subfolders have multiple images, they will play in sequence (recommended for animations).

Notes

1. The file must be a 32-bit, ARGB file.

2. Image file names must be comprised of letters, numbers, underscores and dashes only.

3. Transparency is supported via image alpha channel.

4. The particle image files cannot exceed 1024x1024 in size and if they are part of an animation, they must all be the same size.

Value Specifications: Type: FilePath

SoundEffectPath *(Optional)*

Specifies path to a folder containing one or more sound files that will play (randomly if there is more than one sound file) as particles are emitted.

Files must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

SpawnMethod *(Default)*

Defines how the emitted particles are placed relative to the tracking regions. Three options are available:

Replaces the **RandomScatter** attribute.

1. **RANDOM** (Default): Particles will emit from a random location from within the tracking regions.
2. **ORDERED**: Particles within a given tracking region will be evenly spaced, thus avoiding overlapping.
3. **CENTER**: Particles will emit from the center of the tracking regions.

Value Specifications: Type: Enumerated; Range: "RANDOM", "ORDERED", "CENTER"; Default: "RANDOM"

Velocity *(Default)*

Sets the direction of initial movement of all particles in pixels per second upon spawning. Two values are available. The first value controls horizontal direction (a positive value denotes direction to the left; a negative value denotes direction to the right). The second value controls vertical direction (a positive value denotes downward direction; a negative value denotes upward direction). The default setting of "0.0, 0.0" denotes no movement.

Value Specifications: Type: Decimal; Range: -2000.0 - 2000.0; Default: "0.0, 0.0"

Volume *(Dependent)*

Sets the volume for the file specified by the optional **SoundEffectPath** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

4.11.2 Sample: ParticlesEmitMotion-Bubbles

Creates a bubble effect over a specified background. One hundred particles will emit at once, each one expanding to a random width and height between 16 and 32 pixels. Height will always be the same as width, since **MaintainAspectRatio** is set to "True". Particles will be visible for between 0.5 and 2 seconds. The particles rise and accelerate straight upward. No color saturation will occur. Up to 10 random sounds in the specified folder will play every 0.05 seconds.

ParticlesEmitMotion1\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesEmitMotion1">
  <ParticlesEmitMotion
    EmitRate="100"
    MinWidth="16"
    MaxWidth="32"
    MaintainAspectRatio="True"
    MinLiveTime="0.5"
    MaxLiveTime="2.0"
    FadeTime="0.5"
    Velocity="0.0, -50.0"
    Acceleration="0.0, -150.0"
    ExpandRate="1.0"
    ParticleTexturePath=".\Images\bubble.tga"
    BlendMode="normal"
    SoundEffectPath=".\Sounds\bubble"
    Volume="1.0"
    MaxSoundSamples="10"
    MinSoundDelay="0.05"
    MaxSoundDelay="0.05"
    >
      <Image Name=".\Images\bg_blue.jpg"/>
  </ParticlesEmitMotion>
</AdTracker>
```

4.11.3 Sample: ParticlesEmitMotion-Fire

Creates a fire effect over a specified background. Many particles will emit at once, each expanding to a random width between 96 and 128 pixels. Height will always be the same as width, since **MaintainAspectRatio** is set to "True". Particles will be visible for between 0.0 and 3.0 seconds. The particles flow downward and to the left. No color saturation will occur. In order for the effect to occur only in a desired region, it has been placed within an **AlphaMask** play item.

ParticlesEmitMotion2\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesEmitMotion2">
  <AlphaMask
    Duration="10"
    AlphaImage = ".\Images\alpha_Pacaya.tga"
    >
      <ParticlesEmitMotion
        Duration="10"
        EmitRate="768"
        MinWidth="96"
        MaxWidth="128"
        MaintainAspectRatio="True"
        MinLiveTime="0.0"
        MaxLiveTime="3.0"
        FadeTime="1.0"
        Velocity="0.0, -32.0"
        Acceleration="0.0, -64.0"
        ExpandRate="0.8"
        ParticleTexturePath=".\Images\star_red.tga"
        >
          <Image Name=".\Images\Pacaya.jpg"/>
      </ParticlesEmitMotion>
  </AlphaMask>
</AdTracker>
```

4.12 ParticlesImageSwitch

This is an **Effect** play item that displays a particle grid. One set of particles is displayed when motion is not detected. Upon motion detection a different set will be displayed in their place.

The particle image files cannot exceed 1024x1024 in size and they must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

ParticlesImageSwitch works with or without a child play item.

4.12.1 Attributes

ActiveTexturePath *(Required)*

Specifies the path to the folder which contains one or more of the image files upon which the active particles are based. The active particles are those that are displayed when interaction is detected. Alpha channel is supported.

Notes

1. The particle image files cannot exceed 1024x1024 in size and image files in the same folder must all be the same size.
2. Images must be RGB or RGBA format files.
3. Image file names must be comprised of letters, numbers, underscores and dashes only.
4. The number of images in each texture path folder should be the same. If not, then the number of images used will be the lesser of the two.
5. Image file names must be comprised of letters, numbers, underscores and dashes only.
6. The **ActiveTexturePath** images that appear upon interaction are mapped to a corresponding **InactiveTexturePath** image based on the order of the images in the folder. The images are sorted based on their filenames according to standard Windows® alphanumeric sorting conventions.

Example

Files would be sorted as follows:

- file01.img
- file10.img
- file11.img
- file2.img
- file4.img

Value Specifications: Type: FilePath

GridWidth, GridHeight *(Default)*

The number of displayed particles are based on these two attributes. **GridWidth** specifies the number of particles across. **GridHeight** specifies the number of particles down. The size of each particle is based on the total number of particles on the screen. The greater the number of particles, the smaller the size of each particle.

Value Specifications: Type: Integer; Range: 1 - 128; Default: GridWidth: 16, GridHeight: 12

InactiveTexturePath *(Required)*

Specifies the path to the folder which contains one of more image files upon which the inactive particles are based. The inactive particles are those that are displayed when no interaction is detected. Alpha channel is supported.

Notes

1. The particle image files cannot exceed 1024x1024 in size and image files in the same folder must all be the same size.
2. Images must be RGB or RGBA format files.
3. Image file names must be comprised of letters, numbers, underscores and dashes only.
4. The number of images in each texture path folder should be the same. If not, then the number of images used will be the lesser of the two.
5. The **ActiveTexturePath** images that appear upon interaction are mapped to a corresponding **InactiveTexturePath** image based on the order of the images in each folder. The images are sorted based on their filenames according to standard Windows® alphanumeric sorting conventions.

   Example

   Files would be sorted as follows:

   - file01.img
   - file10.img
   - file11.img
   - file2.img
   - file4.img

   Value Specifications: Type: FilePath

TouchTime *(Default)*

The amount of time, in seconds, the active particle will remain on screen following interaction.

Value Specifications: Type: Decimal; Range: Greater than 0.0; Default: 2.0

4.12.2 Sample: ParticlesImageSwitch

Creates a 16 X 12 particle grid that switches particle images upon interaction.

ParticlesImageSwitch\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesImageSwitch">
  <ParticlesImageSwitch
    InactiveTexturePath=".\Images\Inactive"
    ActiveTexturePath=".\Images\Active"
    GridWidth="16"
    GridHeight="12"
    TouchTime="1.0"
    >
  </ParticlesImageSwitch>
</AdTracker>
```

4.13 ParticlesInteract

This **Effect** play item is no longer supported. It is recommended that ParticlesInteractAnimate 97 should be used even when using non-animated particles.

This play item causes particles to react to motion detection. Attributes control such things as the amount and appearance of particles as well as how they move.

The particle image files cannot exceed 1024x1024 in size and they must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

4.13.1 Attributes

This play item is no longer supported. It is recommended that ParticlesInteractAnimate 97 should be used even when using non-animated particles.

BoundaryImage *(Optional)*

An optional boundary image may be specified in order to limit where the particles will initially appear. If the **EnforceBorders** attribute is set to "True", then the particles will always stay inside those areas set by the boundary image.

Particles will appear in any area in which the boundary image's alpha channel is set to "0" (fully transparent). When the play item begins, it will attempt to place all particles within those areas specified by the boundary image. In some cases particles may appear outside the boundary area.

Notes

1. The boundary image must be a ARGB png or tga file.
2. The boundary image is not visible. It only defines the boundary areas.
3. The dimensions of the boundary image itself (not the desired boundary area) must be the same as the Dazzler screen resolution as specified in Dazzler Setup 3. If not, then the boundary image will be ignored.

Value Specifications: Type: FilePath

Collisions *(Default)*

Specifies whether or not particles will collide with each other.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

EnableRotation *(Default)*

Controls whether particles rotate upon motion detection or collision.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

EnforceBorders *(Default)*

When set to "True", the particles will collide with boundaries, including screen borders as well as the borders of a boundary image, if specified. When set to "False", all borders are ignored, which also means particles can leave the screen area entirely. When a particle completely leaves the screen area, they will not reappear.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

KineticFriction *(Default)*

Controls the amount of resistance applied to particles upon movement. A value of "0.0" specifies no resistance.

Value Specifications: Type: Decimal; Range: 0.0 - 100; Default: 5.0

MaxForce (*Default*)

Controls the amount of force applied to a particle upon motion detection or collision. The higher the value, the more force is applied.

Force is accumulative. When continuous, non-stop interaction is applied to a particle, the amount of force will increase. For example, pushing, instead of hitting, a particle will increase the force applied to it.

Value Specifications: Type: Decimal; Range: 0.0 - 100; Default: 10.0

MaxSounds *(Dependent)*

When sounds are specified, controls the maximum number of concurrent sound effects.

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 16

ParticleCount *(Default)*

Specifies the number of particles to display. Because precedence is given to the even distribution of particle images, if the **ParticleCount** is not a multiple of the number of particle images available, the actual number of particles displayed will be adjusted to ensure equal distribution of images.

When the Collision attribute is set to "True", a high **ParticleCount** value relative to the available screen area could return unexpected results.

Value Specifications: Type: Integer; Range: 1 - 16383; Default: 250

ParticleTexturePath *(Required)*

Specifies the path to the folder which contains the image files upon which the particle image is based.

Notes

1. Path must specify a folder only. Whether using one or more image files, place them in the specified folder.

2. The particle image files cannot exceed 1024x1024 in size and image files in the same folder must all be the same size.

3. The files must be 32-bit, RGB.

4. Image file names must be comprised of letters, numbers, underscores and dashes only.

5. For best results, it is recommended that image files be created square.

6. Transparency is supported via image alpha channel.

Value Specifications: Type: FilePath

ParticleRadius *(Default)*

Sets a collision radius around each particle. This controls the points around the particle that react to motion detection and, depending on other relevant attribute settings, to boundary and particle collisions as well. In most cases the radius should be set to surround the particle texture image as closely as possible.

A value of "0.0" attempts to calculate a radius based on the **ParticleWidth** and **ParticleHeight** values.

If the dimensions of a particle texture are considerably smaller than the dimensions of the image file it is part of, it is recommended to enter a value that is as precise as possible.

Value Specifications: Type: Decimal; Range: 0.0 - 1024; Default: 0.0

ParticleWidth, **ParticleHeight** *(Default)*

Sets, in pixels, the width and height of the particle image.

The **ParticleWidth** and **ParticleHeight** settings control the dimensions of the particle texture itself on the screen. Your provided images will be scaled to fit this texture. If your image has borders around it in your file, then these borders will show up within the dimensions of the particle texture. For best results, you should minimize any borders or "blank areas" around the perimeter of your images within your image files.

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 64

Even though it is recommended that the images should be created square, the particle dimensions need not be.

SoundEffect *(Optional)*

If **Collisions** is set to "True", specifies path to a sound file that plays whenever two particles collide.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Volume *(Dependent)*

Sets the volume for the file specified by the optional **SoundEffect** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

4.13.2 Sample: ParticlesInteract - Sweeping Rocks

Creates an interactive particle effect of 256 rock-like particles displayed over an image play item. The particles are 64 pixels square and will bounce of the presentation area borders. The particles will rotate. A particle radius of 24 pixels has been set.

ParticlesInteract\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesInteract">
  <ParticlesInteract
    ParticleCount="256"
    ParticleWidth="64"
    ParticleHeight="64"
    ParticleRadius="24"
    ParticleTexturePath=".\Images\rocks"
    KineticFriction="2.8"
    MaxForce="10"
    Collisions="False"
    EnableRotation="True"
    EnforceBorders="True"
    >
      <Image Name=".\Images\bg_moon.jpg"/>
  </ParticlesInteract>
</AdTracker>
```

4.14 ParticlesInteractAnimate

This is an **Effect** play item that causes onscreen animated or non-animated elements (particles) to react to motion detection. The animating of this play item is not a reference to screen movement, which is common to all particle play items, but rather to how the particle itself is an animation. Attributes control the amount of particles and their behavior. The specified number of particles appear on-screen as soon as the effect begins and are displayed over appropriate child play items.

The particle image files cannot exceed 1024x1024 in size and they must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

ScreenXtreme Users! See the special **ParticlesInteractAnimate** samples using the **KeyedCameraViewer**:

ParticlesInteractAnimate with Video 153

Popcorn with Video 157

4.14.1 Attributes

AbsoluteAngle*n* (Dependent)

If a corresponding particle uses the **AbsolutePosition***n* attribute, this attribute allows for the setting of the angle in which the particle will move upon interaction. Angle is calculated in degrees with zero (0) set at the 3 o'clock position and increasing in a clockwise direction.

Value Specifications: Type: Decimal; Range: 0.0 - 360.0; Default: 0.0

AbsolutePosition*n* (Optional)

Allows you to specify exact positions for particles. You may list absolute positions for as many particles as you like. If more particles are displayed than have absolute positions specified, the remaining particles will be positioned randomly. The desired position is specified by using x,y coordinates in pixels, where "x" is the horizontal position from the left, and "y" is the vertical from the top. The position specified by the coordinates indicates the center of the particle. The particles will be layered in order, starting with the first particle closest to the background.

Notes

- If no absolute position is specified, then all particles will be positioned randomly.
- This attribute is specified by stating its name immediately followed by a number in sequence, beginning with "1" (even when only specifying absolute position for only one particle). For an example, see the ParticlesInteractAnimate - Flowers 110 sample.

Value Specifications: Type: Integer; Range: 1 - screen resolution

AbsoluteVelocity*n* (Dependent)

If a corresponding particle uses the **AbsolutePosition*n*** attribute (which in turn uses the **AbsolutePosition*n*** attribute), this attribute allows for the setting of the speed, in pixels per second, at which the particle will move upon interaction.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 0.0

Acceleration (Default)

Sets the rate of movement, in pixels per second, applied to all particles upon spawning. Two values are available. The first value controls horizontal direction (a positive value denotes direction to the left; a negative value denotes direction to the right). The second value controls vertical direction (a positive value denotes downward direction; a negative value denotes upward direction). The default setting of "0.0, 0.0" denotes no acceleration.

Value Specifications: Type: Decimal; Range: -5000 - 5000; Default: "0.0, 0.0"

For best results when using **Acceleration**, avoid using the **Reset** attribute.

AnimationCycleTime (Default)

Time, in seconds, it takes to complete one animation sequence. The higher the number, the slower the sequence.

Value Specifications: Type: Decimal; Range: Greater than 0.0; Default: 1.0

AnimationMinDelay, **AnimationMaxDelay** (Dependent)

If **AnimationMode** is set to "TWOSHOTDELAY", specifies the minimum and maximum amount of time, in seconds, before the animation reverses. A random amount of time between the two settings will be selected for each particle. **AnimationMaxDelay** must be equal to or greater than **AnimationMinDelay**.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: **AnimationMinDelay**: 0.5 / **AnimationMaxDelay**: 2.0

These attributes are only available when **AnimationMode** is set to "TWOSHOTDELAY".

AnimationMode (Default)

Controls animation behavior as follows:

1. LOOP: After playing the last image in the sequence, the animation will return to its first image and continue indefinitely. Animation runs as soon as play item starts.

2. ONESHOT: The animation sequence will play through its images only once and stop at the last image. This also applies when **RandomStartFrame** is set to "True". In that case whatever frame the sequence begins with, the animation will still stop with the last one in the folder. If the random image happens to be the last one in folder, that particle will not animate at all. Animation runs as soon as play item starts.

3. PINGPONG: The animation sequence will continuously alternate between playing its images forward and reverse. Animation runs as soon as play item starts.

4. TOUCHLOOPRESET: Animation plays upon interaction. The animation will continue to play until the particle stops moving.

5. TOUCHONESHOTRESET: Upon interaction, the animation plays once through and stops when it reaches the first image again. It will play each time it is touched.

6. TWOSHOTDELAY: Upon interaction, the animation sequence with play through its images and then pause based on the time set by the AnimationMinDelay/MaxDelay attributes. It will then play through its images in reverse. The animation stops until interaction is detected again.

The behavior of the **TouchSound** attribute is dependent on which animation mode is being used. See the **TouchSound** attribute for more information.

Value Specifications: Type: Enumerated; Options: "LOOP", "ONESHOT, "PINGPONG", "TOUCHLOOPRESET", "TOUCHONESHOTRESET" or "TWOSHOTDELAY"; Default: "LOOP"

BlendMode *(Default)*

Controls the blending of overlapping particles . "Normal" prevents color bleeding and saturation.

Value Specifications: Type: Enumerated; Options: "Normal" or "Saturate"; Default: "Normal"

BoundaryHitSound *(Optional)*

If **BoundaryImage** is active (**BoundaryImage** path is specified, and **EnforceBorders** is set to "True"), specifies path to a sound file that plays whenever a particle collides with a boundary.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

BoundaryHitVolume *(Dependent)*

Sets the volume for the file specified by the optional **BoundaryHitSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

BoundaryImage *(Optional)*

An optional boundary image may be specified in order to limit where the particles will initially appear. If the **EnforceBorders** attribute is set to "True", then the particles will always stay inside those areas set by the boundary image.

Particles will appear in any area in which the boundary image's alpha channel is set to "0" (fully transparent). When the play item begins, it will attempt to place all particles within those areas specified by the boundary image. In some cases particles may appear outside the boundary area.

Notes

1. The boundary image must be a ARGB png or tga file.
2. The boundary image file name must be comprised of letters, numbers, underscores and dashes only
3. The boundary image is not visible. It only defines the boundary areas.
4. The dimensions of the boundary image itself (not the desired boundary area) must be the same as the Dazzler screen resolution as specified in Dazzler Setup. If not, then the boundary image will be ignored.
5. When using a boundary image, the **ParticleRadius** attribute is used to help organize the placement of the particles. For best results, it is recommended to use a relatively small **ParticleRadius** value (10 or less).

Value Specifications: Type: FilePath

Collisions *(Default)*

Specifies whether or not particles will collide with each other.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

CollisionSound *(Optional)*

If **Collisions** is set to "True", specifies path to a sound file that plays whenever two particles collide.

File must be Windows®-compliant WAV.

Value Specifications: Type: FilePath

CollisionVolume *(Dependent)*

Sets the volume for the file specified by the optional **CollisionSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

DynamicRotationFactor (Dependent)

If **EnableRotation** is set to "True" and **RotationSpeed** is set to "0" (this combination of settings causes the particles to only rotate upon interaction), then the rotation speed will be adjusted by the specified factor. For example, a factor of "2.0" will cause the particles to rotate twice as fast, while a factor of "0.5" causes the particles to rotate at half speed. A default setting of "1.0" causes no change.

Value Specifications: Type: Decimal; Range: 0.0 - 100.0; Default: 1.0

EnableRotation (Default)

Controls whether particles rotate upon motion detection or collision. If the **RotationSpeed** attribute is set to "0.0", then particles will rotate only upon collision. See the **RotationSpeed** attribute for more information.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

EnforceBorders *(Default)*

When set to "True", the particles will collide with boundaries, including screen borders as well as the borders of a boundary image, if specified. When set to "False", no border collisions will occur, which also means particles can leave the screen area entirely. When a particle completely leaves the screen area, they will not reappear unless the **Reset** attribute is set to "True".

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

KineticFriction (Default)

Controls the amount of resistance applied to particles upon movement. A value of "0.0" specifies no resistance.

Value Specifications: Type: Decimal; Range: 0.0 - 100.0; Default: 5.0

MaxForce (Default)

Controls the amount of force applied to a particle upon motion detection or collision. The higher the value, the more force is applied.

Force is cumulative. When continuous, non-stop interaction is applied to a particle, the amount of force increases. For example, pushing instead of hitting a particle will increase the force applied to it.

Value Specifications: Type: Decimal; Range: 0.0 - 100.0; Default: 10.0

MaxSounds *(Dependent)*

When sounds are specified, controls the maximum number of concurrent sound effects.

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 16

NonAnimatedParticlePath (*Required*)

Specifies the path to a folder which contains separate one-frame animations for every image file.

Value Specifications: Type: FilePath

ParticleCount (Default)

Specifies the number of particles to display. Precedence is given to the even distribution of particle images, so if the **ParticleCount** is not a multiple of the number of particle images available, the actual number of particles displayed will be adjusted to ensure equal distribution of image animation sequences.

Value Specifications: Type: Integer; Range: 1 - 16383; Default: 250

While 16348 is technically the particle count limit, the actual allowable number of particles depends on system resources. Setting the **Collision** attribute is set to "True" is an additional stress on the particle count. A high **ParticleCount** value relative to the available screen area could return unexpected results.

ParticleRadius (Default)

Sets a collision radius around each particle. This controls the points around the particle that react to motion detection and, depending on other relevant attribute settings, to boundary and particle collisions as well. In most cases the radius should be set to surround the particle texture image as closely as possible.

Notes

1. A value of "0.0" attempts to automatically calculate a radius based on the **ParticleWidth** and **ParticleHeight** values.

2. If the dimensions of a particle texture are considerably smaller than the dimensions of the image file it is part of, it is recommended to enter a value that is as precise as possible.

3. When using **BoundaryImage** and/or when **Collisions** is set to "True", particles will be arranged based on the specified radius. Depending on the radius value and number of particles, the particles may spawn in a grid layout and may stack on top of each other.

Value Specifications: Type: Decimal; Range: 0.0 - 1204.0; Default: 0.0

ParticleTexturePath (Required)

Specifies the path to the folder which contains subfolders which in turn contain the image files upon which the particle image files are based. The animations are created by sequencing through the images in each subfolder. Non-animated particles may be created using the **NonAnimatedParticlePath** attribute.

Notes

1. Path must specify a folder only.

2. Specified folder must have at least one subfolder with images. Do not place any images in the specified folder.

3. More than one subfolder is needed only if more than one animation sequence is desired. This could be because different looking particles are needed or if it is necessary for some of the particles to begin the animation sequence at a different point.

4. The particle image files cannot exceed 1024x1024 in size and image files in the same folder must all be the same size.

5. Image file names must be comprised of letters, numbers, underscores and dashes only.

6. The order of the animation sequence is based on the file names of the images according to standard Windows® alphanumeric sorting conventions.

7. Transparency is supported via image alpha channel.

Example

Files would be sorted as follows:

- file01.img
- file10.img
- file11.img
- file2.img
- file4.img

Value Specifications: Type: FilePath

ParticleWidth, ParticleHeight (Default)

Sets, in pixels, the width and height of the particle image.

The **ParticleWidth** and **ParticleHeight** settings control the dimensions of the particle texture itself on the screen. Your provided images will be scaled to fit this texture. If your image has borders around it in your file, then these borders will show up within the dimensions of the particle texture. For best results, you should minimize any borders or "blank areas" around the perimeter of your images within your image files.

Image file names must be comprised of letters, numbers, underscores and dashes only

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 64

RandomStartAngle (Default)

If set to "True", particles will appear at different angles.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

RandomStartFrame (Default)

When set to "True", animations begin with a random image in the sequence and continue in order. This is helpful when you want similar animations displayed, but to not be in sync with one another.

When **RandomStartFrame** is set to "True" and using one of the following animation modes, PINGPONG, TWOSHOTDELAY, TOUCHLOOPRESET, or TOUCHONESHOTRESET, the animation will start with a random image, but continue as normal. As the animation plays, the random starting image will not be treated as if it is the first image of the sequence.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

Reset *(Default)*

If set to "True", the particles will attempt to return to their starting positions.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False

For best results when using **Reset**, avoid using the **Acceleration** attribute.

When setting **Reset** to "True", avoid low **KineticFriction** settings, for it may result in undesirable results.

ResetForce *(Dependent)*

If **Reset** is set to "True", specifies the amount of force applied to particles once they are in motion as they attempt to return to their original positions.

Value Specifications: Type: Decimal; Range: 0.0 - 100.0; Default: 10.0

A setting of "0.0" will cause the particles to only move upon interaction. They will not attempt to return to their original positions.

RotateToDirection (Dependent)

If **EnableRotation** is set to "True", then setting this attribute to "True" will rotate Particles in the direction they are moving.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

RotationSpeed (Dependant)

If **EnableRotation** is set to "True", the speed, in degrees per second, at which the particles will rotate. A positive value indicates a counterclockwise direction; a negative value indicates a clockwise direction. A setting of "0.0" will cause the particles to rotate only when a collision of any kind occurs.

Value Specifications: Type: Decimal; Range: -2880.0 - 2880.0; Default: 0.0

TouchLoopTolerance (Dependent)

When using the TOUCHLOOPRESET animation mode, allows for the animation to stop when motion stops. At times motion may be so slow as to be unperceptive to the eye. This attribute sets a tolerance factor that will cause the animation to stop before the particle itself completely stops moving.The lower the value, the longer the animation will continue to play.

Value Specifications: Type: Decimal; Range: Greater than 0.0; Default: 1.0

TouchSound *(Optional)*

Specifies path to a sound file that plays when a particle intersects with the tracker outline. The behavior of the sound depends on which animation mode is being used. For more information on the animation modes themselves, see the **AnimationMode** attribute.

LOOP - Sound plays every time a particle intersects with the tracker outline.

ONESHOT - Sound plays every time a particle intersects with the tracker outline.

PINGPONG - Sound plays every time a particle intersects with the tracker outline.

TWOSHOTDELAY - Sound plays only when the particle intersects with the tracker outline if the animation sequence is at the first image. Otherwise intersecting with the tracker outline will not produce a sound.

TOUCHLOOPRESET - Once the particle is touched, sound will play each time the animation sequence loops back to the first frame.

TOUCHONESHOTRESET - Sound plays when the particle intersects with the tracker outline only if the animation sequence is at the first image.

Notes

- File must be Windows®-compliant WAV.
- When using an animation mode that only plays a sound upon the first image of the sequence, if **RandomStartFrame** is set to "True", the sound will play upon the random start frame the first time through the sequence, but only on the first frame of the sequence after that.

Value Specifications: Type: FilePath

TouchVolume *(Dependent)*

Sets the volume for the file specified by the optional **TouchSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

WallHitSound *(Optional)*

If **EnforceBorders** is set to "True", specifies path to a sound file that plays whenever a particle collides with the screen boundary.

File must be Windows®-compliant WAV.

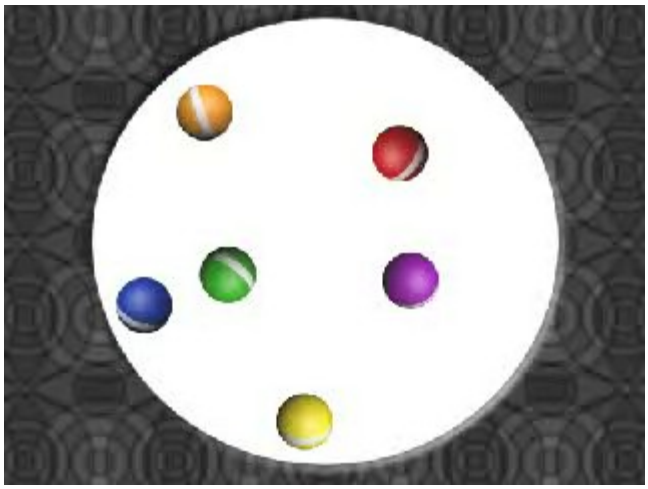
Value Specifications: Type: FilePath

WallHitVolume *(Dependent)*

Sets the volume for the file specified by the optional **WallHitSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

4.14.2 Sample: ParticlesInteractAnimate - Boundary Image & Sound



Creates an animated interactive particle effect of 6 particles within a boundary image. Particles are 96 pixels in both width and height. The particles collide with each other and the boundary image border. Two different sounds are used, one when the particles collide with each other and one when they collide with the boundary image border. The particles rotate only upon interaction.

ParticlesInteractAnimate2\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesInteractAnimate1">
  <ParticlesInteractAnimate
    ParticleCount="6"
    ParticleWidth="96"
    ParticleHeight="96"
    ParticleRadius="45"
    ParticleTexturePath=".\Images\balls"
    KineticFriction="2.0"
    MaxForce="2"
    BoundaryImage=".\Images\mymask.tga"
    EnforceBorders="True"
    BlendMode="Normal"
    EnableRotation="True"
    RotationSpeed="0.0"
    Collisions="True"
    AnimationCycleTime="1.0"
    AnimationMode="LOOP"
    RandomStartFrame="True"
    MaxSounds="2"
    CollisionSound=".\Sounds\ping.wav"
    CollisionVolume="0.5"
    BoundaryHitSound=".\Sounds\tap.wav"
    BoundaryHitVolume="0.5"
    >
      <Image Name=".\Images\mymask.tga"/>
  </ParticlesInteractAnimate>
</AdTracker>
```

4.14.3 Sample: ParticlesInteractAnimate - Collision & Sound

Creates an animated interactive particle effect of 18 particles that displays over an image play item. The particles are 64 pixels square, collide with one another, and bounce off the presentation area borders. The particles rotate. A particle radius of 30 pixels has been set. The animation sequences loop continuously and begin at a random frame in the sequence. Only collision sound is enabled. Up to 6 sounds can play simultaneously.

ParticlesInteractAnimate1\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesInteractAnimate2">
  <ParticlesInteractAnimate
    ParticleCount="18"
    ParticleWidth="64"
    ParticleHeight="64"
    ParticleRadius="30"
    ParticleTexturePath=".\Images\balls"
    KineticFriction="2.0"
    MaxForce="10"
    EnforceBorders="true"
    BlendMode="normal"
    EnableRotation="true"
    RotationSpeed="0.0"
    Collisions="true"
    AnimationCycleTime="1.0"
    AnimationMode="LOOP"
    RandomStartFrame="true"
    MaxSounds="6"
    CollisionSound=".\Sounds\ping.wav"
    CollisionVolume="0.4"
    >
      <Image Name=".\Images\logo.jpg"/>
  </ParticlesInteractAnimate>
</AdTracker>
```

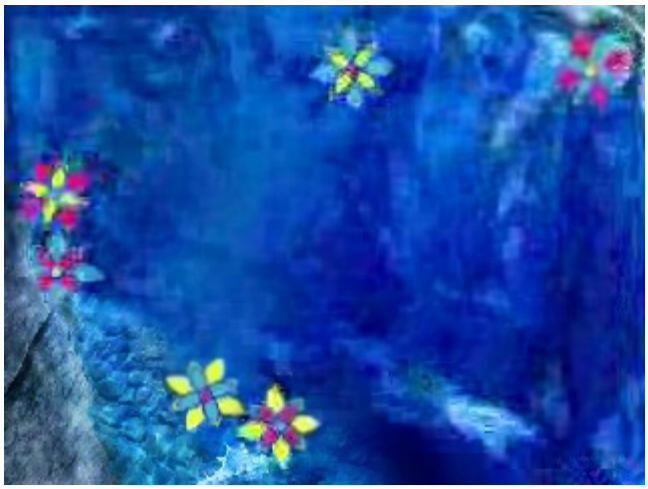
4.14.4 Sample: ParticlesInteractAnimate - Popcorn

Displays popcorn kernels within a set boundary and appear at random angles. Kernels become popcorn upon interaction. Particles can scatter beyond boundary, but will reset to their original position. They will also revert to kernels anywhere from 6 and 10 seconds inclusive. The particles do not collide with one another. The rotation factor is set to 60% to slow down the rotation speed of each particle. No sounds are used.

ParticlesInteractAnimate3\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesInteractAnimate3">
  <ParticlesInteractAnimate
    ParticleTexturePath=".\Images\popcorn"
    ParticleCount="850"
    ParticleWidth="64"
    ParticleHeight="64"
    ParticleRadius="10"
    MaxForce="10.0"
    KineticFriction="8.0"
    RandomStartAngle="True"
    EnableRotation="True"
    RotationSpeed="0.0"
    DynamicRotationFactor="0.6"
    RandomStartFrame="False"
    AnimationCycleTime="0.5"
    AnimationMode="TWOSHOTDELAY"
    AnimationMinDelay="6.0"
    AnimationMaxDelay="10.0"
    BlendMode="Normal"
    Reset="True"
    ResetForce="5"
    BoundaryImage=".\Images\popcorn_boundary.tga"
    EnforceBorders="False"
    Collisions="False"
    >
      <Image Name=".\Images\popcorn_bg.tga"/>
  </ParticlesInteractAnimate>
</AdTracker>
```

4.14.5 Sample: ParticlesInteractAnimate - Flowers



This sample uses **ParticlesInteractAnimate** to display several colored flowers that rotate upon interaction and will return to their start positions. The flowers themselves are not animated. They collide with the screen boundary, but not with each other. Absolute positioning is specified for all particles.

ParticlesInteractAnimate4\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesInteractAnimate4">
  <ParticlesInteractAnimate
    ParticleCount="6"
    AbsolutePosition1="100,400"
    AbsolutePosition4="200,230"
    AbsolutePosition3="320,620"
    AbsolutePosition5="440,840"
    AbsolutePosition2="610,210"
    AbsolutePosition6="950,250"
    ParticleWidth="148"
    ParticleHeight="148"
    ParticleTexturePath=".\Images\flowers"
    KineticFriction="3.4"
    MaxForce="2"
    BlendMode="Normal"
    EnableRotation="True"
    RotationSpeed="0.0"
    RotationFactor="2.0"
    RandomStartFrame="False"
    EnforceBorders="True"
    Collisions="False"
    Reset="True"
    ResetForce="3.0"
    >
      <Include Name=".\Water\root.spark"/>
  </ParticlesInteractAnimate>
</AdTracker>
```

4.15 ParticlesInteractVapor

This is an **Effect** play item that causes particles, upon interaction, to move and to fade out.

Once particles are completely faded and/or off screen, they will not reappear.

The particle image files cannot exceed 1024x1024 in size and they must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

4.15.1 Attributes

AutoFadeoutDuration *(Dependent)*

Specifies, in seconds, how long automatic fade out will take, if and when **AutoFadeoutStart** begins.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 2.0

Setting the value to "0.0" sets the attribute to its default value. If instant fade out is desired, set the value to "0.01".

AutoFadeoutStart *(Default)*

Specifies the amount of seconds, prior to the termination of the play item, that all remaining particles will begin to automatically fade out.

Value Specifications: Type: Decimal; Range: Greater than 0.0; Default: 0.0

This attribute should only be used when ParticlesInteractVapor is inside SequentialScheduler 39,

The default value of "0.0" effectively turns this attribute off.

BlendMode *(Default)*

Controls the blending of overlapping particles . "Normal" prevents color bleeding and saturation.

Value Specifications: Type: Enumerated; Options: "Normal" or "Saturate"; Default: "Saturate"

FadeoutDuration *(Default)*

How long in seconds it will take for a particle to fade out once fade out begins. The higher the value, the longer the fade out sequence will be.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 1.0

FadeoutForce *(Default)*

Controls the amount of force needed to apply to the particles before fade out begins. The higher the value, the greater the amount of force is required before fade out begins.

Value Specifications: Type: Decimal; Range: 0.0 - 1000.0; Default: 0.01

KineticFriction *(Default)*

Controls the amount of resistance applied to particles upon movement. A value of "0.0" specifies no resistance.

Value Specifications: Type: Decimal; Range: 0.0 - 100.0; Default: 2.0

MaxForce *(Default)*

Controls the amount of force applied to a particle upon motion detection. The higher the value, the more force is applied.

Force is cumulative. When continuous, non-stop interaction is applied to a particle, the amount of force increases. For example, pushing, instead of hitting, a particle will increase the force applied to it.

Value Specifications: Type: Decimal; Range: 1.0 - 1000.0; Default: 0.6

ParticleCount *(Default)*

Specifies the number of particles to display. Because precedence is given to the even distribution of particle images, if the **ParticleCount** is not a multiple of the number of particle images available, the actual number of particles displayed will be adjusted to ensure equal distribution of images.

Value Specifications: Type: Integer; Range: 1 - 16383; Default: 500

ParticleTexturePath *(Required)*

Specifies the path to the folder which contains the image files upon which the particle image is based.

Notes

1. The files must be 32-bit, RGB.
2. Path must specify a folder only. Whether using one or more image files, place them in the specified folder.
3. Image file names must be comprised of letters, numbers, underscores and dashes only.
4. The particle image files cannot exceed 1024x1024 in size and image files in the same folder must all be the same size.
5. Transparency is supported via image alpha channel.

Value Specifications: Type: FilePath

ParticleWidth, ParticleHeight *(Default)*

Sets, in pixels, the width and height of the particle image.

Value Specifications: Type: Integer; Range: 1 - 1024; Default: 128

RotationSpeed *(Default)*

Specifies a maximum rotation speed for each particle in degrees per second. A random speed in either direction will be selected up to the value specified. Rotation begins as soon as play item starts. A value of "0.0" specifies no rotation.

Setting a rotation speed provides a realistic, animated look to vapor-like particles.

Value Specifications: Type: Decimal; Range: -1440.0 - 1440.0; Default: 1.0

4.15.2 Sample: Sweep Away Gas Particles

Creates a gas-like effect over an Image play item. 1500 particles of 128 x 128 each will appear as soon as the effect begins.

ParticlesInteractVapor\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesInteractVapor">
  <ParticlesInteractVapor
    ParticleCount="1500"
    ParticleWidth="128"
    ParticleHeight="128"
    ParticleTexturePath=".\Images\Gas"
    KineticFriction="3"
    MaxForce="5.0"
    FadeoutForce="0.1"
    FadeoutDuration="0.4"
    RotationSpeed="180.0"
    >
      <Image Name=".\Images\moon.jpg"/>
  </ParticlesInteractVapor>
</AdTracker>
```

4.16 ParticlesReset

This is an **Effect** play item that displays an interactive particle grid. The particles move upon interaction and then begin to return to their original location.

The particle image files cannot exceed 1024x1024 in size and they must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

ParticlesReset works with or without a child play item.

ScreenXtreme Users! See the special **ParticlesReset** sample using the **KeyedCameraViewer**:

ParticlesReset with Video 156

4.16.1 Attributes

GridWidth, GridHeight *(Default)*

The number of displayed particles are based on these two attributes. **GridWidth** specifies the number of particles across. **GridHeight** specifies the number of particles down. The size of each particle is based on the total number of particles on the screen (See also the **TopLeft** and **BottomRight** attributes). The greater the number of particles, the smaller the size of each particle.

Value Specifications: Type: Integer; Range: 1 - 128; Default: GridWidth: 16, GridHeight: 12

InteractionForce *(Default)*

Controls the amount of force applied to each particle when moving. The higher the value, the more force is applied.

Force is accumulative. When continuous, non-stop interaction is applied to a particle, the amount of force will increase. For example, pushing, instead of hitting, a particle will increase the force applied to it.

Value Specifications: Type: Decimal; Range: 1.0 - 1000.0; Default: 10.0

KineticFriction *(Default)*

Controls the amount of resistance applied to particles upon movement. A value of "0.0" specifies no resistance.

Value Specifications: Type: Decimal; Range: 1.0 - 100.0; Default: 1.0

MotionBlur *(Default)*

Controls the amount of blur applied to particles upon interaction.

Value Specifications: Type: Decimal; Range: 0.1 - 0.9; Default: 0.1

ParticleRadius *(Default)*

Sets collision points around each particle that react to motion detection. In most cases the radius should be set to surround the particle texture image as closely as possible.

Value Specifications: Type: Decimal; Range: 0.0 - 1024.0; Default: 0.0

A value of "0.0" attempts to calculate a best radius based on the dimensions of the particles.

If the dimensions of a particle texture are considerably smaller than the dimensions of the image file it is part of, it is recommended to enter a value that is as precise as possible.

ParticleTexturePath *(Required)*

Specifies the path to the folder which contains one of more image files upon which the particles are based.

The particle image files cannot exceed 1024x1024 in size and image files in the same folder must all be the same size.

Image file names must be comprised of letters, numbers, underscores and dashes only.

Images must be 28-bit RGB or 32-bit RGBA files.

Value Specifications: Type: FilePath

ResetForce *(Default)*

Specifies the amount of force applied to particles once they are in motion as they attempt to return to their original positions.

Value Specifications: Type: Decimal; Range: 1.0 - 1000.0; Default: 2.0

A setting of "0.0" will cause the particles to only move upon interaction. They will not attempt to return to their original positions.

TopLeft, BottomRight *(Default)*

Specifies, in pixels, the x and y screen coordinates of the top-left and bottom-right corners of the particle grid. This allows you to create a grid smaller than your entire screen area. The overall grid dimensions will affect particle size. The default settings cause the grid to cover the entire screen.

Value Specifications: Type: Integer; Range: 0 to your maximum screen resolution as specified in your Dazzler configuration [dazzler.cfg]; Default: 0

4.16.2 Sample: ParticlesReset

Creates a 16 X 12 particle grid that will reset following interaction.

ParticlesReset\root.spark

```
<AdTracker AdName="Manual_Samples\ParticlesReset">
  <ParticlesReset
    GridWidth="16"
    GridHeight="12"
    ParticleRadius="0.0"
    ParticleTexturePath=".\Images\ParticlesResetTiles"
    InteractionForce="10"
    KineticFriction="1.0"
    ResetForce="2.0"
    MotionBlur="0.8"
    TopLeft="0.0,0.0"
    BottomRight="0.0,0.0"
    >
      <Image Name=".\Images\earth.jpg" />
  </ParticlesReset>
</AdTracker>
```

4.17 Rotate

This is an **Effect** play item which causes a child play item to rotate.

For best results the display dimensions of the child item should be square.

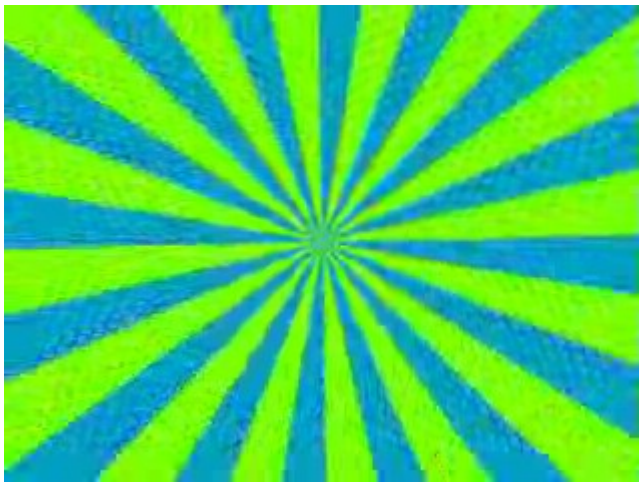
4.17.1 Attributes

RotationSpeed (Required)

Sets the rotation speed and direction in degrees/second. Positive values cause counterclockwise rotation; negative values cause clockwise rotation.

Value Specifications: Type: Decimal; Range: Any positive or negative decimal number; Default: 16.0

4.17.2 Sample: Rotate Image File



Rotates an image play item counterclockwise at 30 degrees per second.

rotate\root.spark

```
<AdTracker AdName="Manual_Samples\Rotate">
  <Rotate
    RotationSpeed="30.0"
    >
      <Image Name=".\Images\pinwheel.tga"/>
  </Rotate>
</AdTracker>
```

4.18 Silhouette

This is an **Effect** play item that displays a Tracker outline silhouette in choice of colors. Supports transparency and the blurring of edges.

To display full silhouette of user presence with or without motion, Background Tracking Mode is recommended (GestureFX only - see GestureFX Tracker documentation for more information).

4.18.1 Attributes

BlurFactor *(Default)*

Controls, in percentage, how much blurring is applied to the edge of the silhouette. The higher the value, the more blur is applied.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.5

Color *(Required)*

An RGBA value that sets the color for the displayed silhouette. Alpha channel (transparency) is supported. For more information about setting color attributes, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0

4.18.2 Sample: Display Silhouette

Displays a 60 percent, opaque, red tracker outline silhouette over an image.

Silhouette\root.spark

```
<AdTracker AdName="Manual_Samples\Silhouette">
  <Silhouette
    Color = "1.0,0.0,0.0,0.6"
    BlurFactor = "0.6"
    >
      <Image Name=".\Images\logo.jpg" />
  </Silhouette>
</AdTracker>
```

4.19 WaterFX

WaterFX is an interactive liquid surface **Effect** play item that applies a realistic ripple or wave to another play item. Motion detection generates a corresponding set of liquid movements. Attributes control how the ripples or waves look and respond.

Suggested uses

This effect is good for giving another play item a watery look. Excellent for creating virtual puddles and lakes or other kind of liquids including gel and oil. Creates interesting results when applied to other effect play items.

This play item requires at least one child play item.

4.19.1 Attributes

Density *(Default)*

Controls the fluidity of the ripple effect, including how quickly the waves die down when there is no further motion detection. A lower value makes the effect appear to settle more quickly, whereas a higher value provides a much more liquid appearance. A value of "0.0001" causes no wave effect at all, while a value of "0.9999" makes the effect appear extremely fluid and take longer to settle.

Value Specifications: Type: Decimal; Range: 0.0001 - 0.9999; Default: 0.98

DensityMap *(Optional)*

Specifying a Density Map image (64 x 64 RGB, 24 or 32-bit greyscale image) allows you to create variable density over the surface of the water effect. The shade of the Density Map image controls which areas of the presentation are affected by the **Density** attribute. The more white the color, the less the effect uses the value set in the **Density** attribute (using pure white will result in no water movement at all). The darker the color, the more the effect uses the value set in the **Density** attribute.

You could create a **DensityMap** image with a gradient running from black to white. This would result in the water effect being more fluid on one end while being more dense on the other.

Density map file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath (64 x 64 RGB, 24 or 32-bit greyscale image)

Height *(Default)*

Controls the amount of height added to the water surface upon motion detection. The higher the value the more pronounced the waves will be and the longer it will take for them to settle.

Value Specifications: Type: Decimal; Range: 0.001 - 5.0; Default: 0.05

InteractionMethod *(Default)*

Specifies one of three methods by which the effect interacts with the user. These methods are:

1. SOLID: The whole interaction area (i.e. the complete silhouette of the user) is used to raise the height of the water.

2. OUTLINE: Only the outline of the interaction area (i.e. the outline of the silhouette of the user) is used to raise the height of the water. This results in less water agitation for the same amount of motion detection than what would be created using "SOLID".

3. SOLID_OUTLINE: While raising the water height around the edges of the outline of the interaction area, there will be no effect at all within the inside of the interaction area. This can be useful when a thicker liquid effect is desired.

Value Specifications: Type: Enumerated; Options: "SOLID", "OUTLINE", or "SOLID_OUTLINE"; Default: "SOLID"

LightColor *(Optional)*

An RGB value that sets a light source color, which enhances the effect by adding additional highlighting. For more information about setting color attributes, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0

The light color may be affected by the color of the child play item. You may need to try several settings before getting the exact light color you want.

LightDirection *(Dependent)*

When the **LightColor** attribute is used, specifies the placement of the light in relationship to the presentation area. Two coordinates are used. The first coordinate (X) sets the horizontal placement with negative values being left of center and positive being right of center. The second coordinate (Y) sets the vertical placement with negative values being below center and positive being above center.

A setting of "0.0" represents the center of the presentation area , while 1.0 and -1.0 are the edges of the presentation area. Interesting light effects can be created by placing the light outside of the presentation area. When doing so, a relatively high **LightPower** attribute value may be required.

Value Specifications: Type: Decimal; Range: Any positive or negative decimal number or zero; Default: "0.0, 0.0"

LightPower *(Dependent)*

When the **LightColor** attribute is used, sets the highlighted area size. The higher the value, the larger the size of the highlight.

Value Specifications: Type: Decimal; Range: 1.0 - 100.0; Default: 100.0

RefractionFactor *(Default)*

Controls the actual look of each wave by how much emphasis is put on the water refracting the child play item. The higher the value, the more distortion is applied when there is a lot of wave activity. The default value of "1.0" produces best results in most cases.

Value Specifications: Type: Decimal; Range: 0.001 - 5.0; Default: 1.0

Speed *(Default)*

Sets how quickly the waves move. The higher the number, the quicker the waves propagate.

Value Specifications: Type: Decimal; Range: 0.001 - 100.0; Default: 20.0

SoundEffect *(Optional)*

Specifies path to a sound file that plays whenever there is motion detection.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

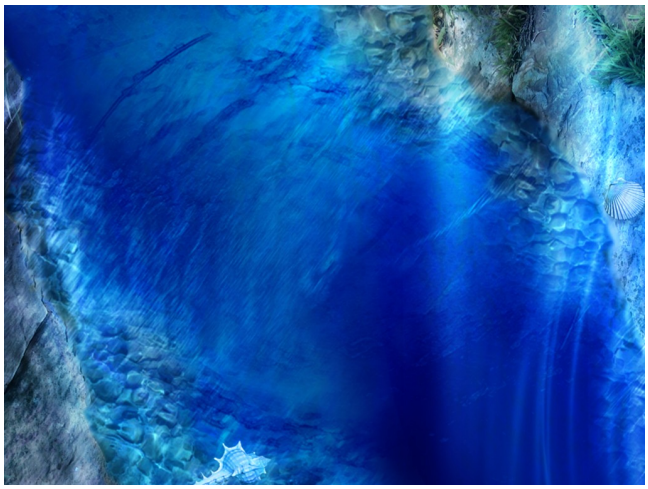
Value Specifications: Type: FilePath

Volume *(Dependent)*

Sets the volume for the file specified by the optional **SoundEffect** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

4.19.2 Sample: WaterFX Normal

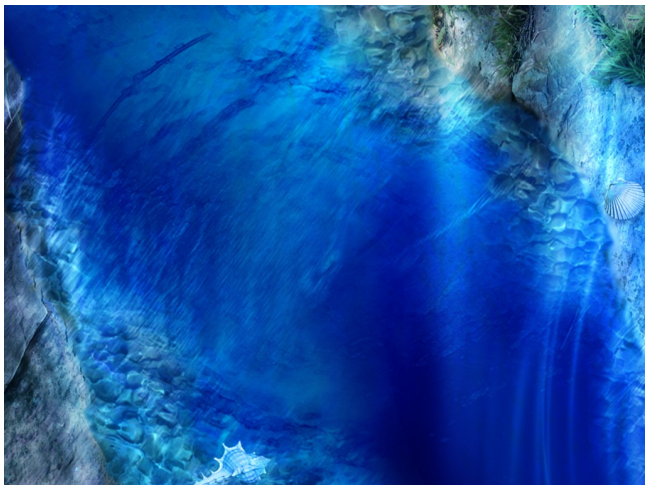


Creates a normal watery effect.

WaterFX_Sample1\root.spark

```
<AdTracker AdName="Manual_Samples\WaterFX_Sample1">
  <WaterFX
    Density="0.98"
    Speed="20.0"
    RefractionFactor="1.0"
    Height="0.05"
    InteractionMethod="Solid"
    >
      <Image Name=".\Images\bg_blue.jpg"/>
  </WaterFX>
</AdTracker>
```

4.19.3 Sample: WaterFX with Density Map and Light

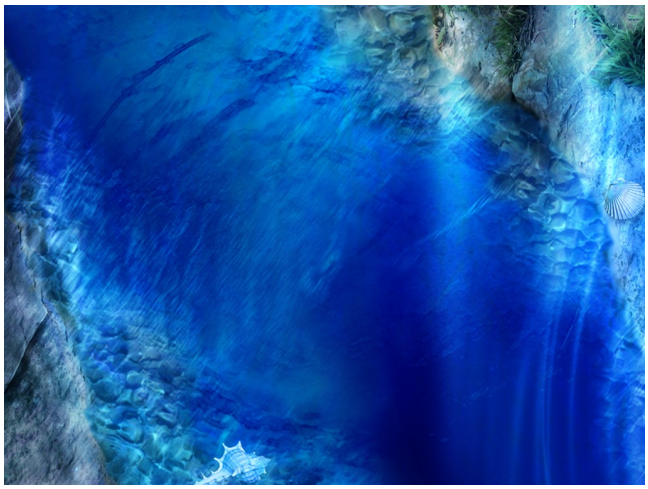


A normal watery effect that is only over part of an image due to the use of a density map. Note that you would need to create and use an actual density map image as described in the **DensityMap** attribute section. A pink light is positioned in the center of the presentation area.

WaterFX_Sample4\root.spark

```
<AdTracker AdName="Manual_Samples\WaterFX_Sample2">
  <WaterFX
    Density="0.98"
    Speed="20.0"
    RefractionFactor="1.0"
    Height="0.05"
    LightColor="1.0, 0.6, 0.0"
    LightPower="40.0"
    LightDirection="0.0, 0.0"
    InteractionMethod="SOLID"
    DensityMap=".\Images\density.tga"
    >
      <Image name=".\Images\bg_blue.jpg"/>
  </WaterFX>
</AdTracker>
```

4.19.4 Sample: WaterFX Normal with Light

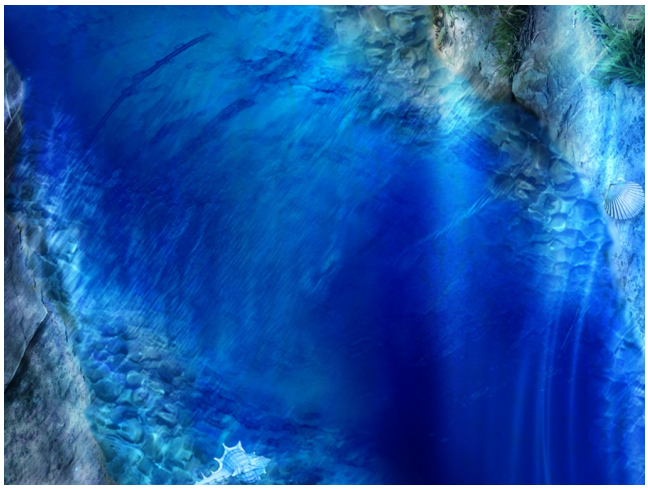


Creates a normal watery effect with a yellow light positioned at the top-right corner of the presentation area.

WaterFX_Sample2\root.spark

```
<AdTracker AdName="Manual_Samples\WaterFX_Sample3">
  <WaterFX
    Density="0.98"
    Speed="20.0"
    RefractionFactor="1.0"
    Height="0.05"
    LightColor="1.0, 1.0, 0.0"
    LightPower="50.0"
    LightDirection="0.9, 0.9"
    InteractionMethod="SOLID"
    >
      <Image name=".\Images\bg_blue.jpg"/>
  </WaterFX>
</AdTracker>
```

4.19.5 Sample: WaterFX Normal with Light and Sound

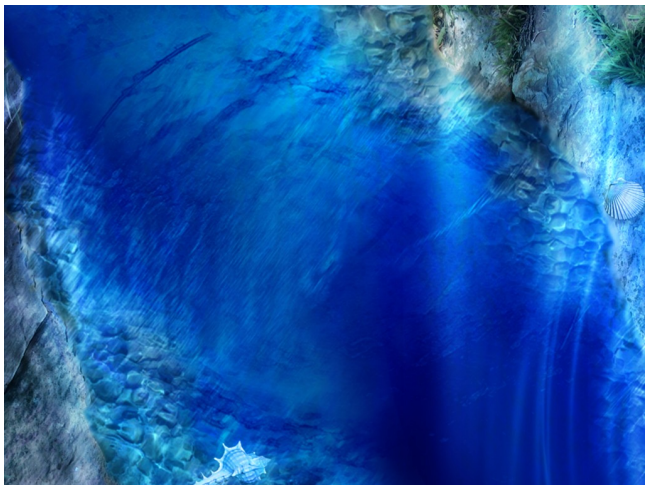


Creates a normal watery effect with sound and a white light positioned at the left-center edge of the presentation area.

WaterFX_Sample3\root.spark

```
<AdTracker AdName="Manual_Samples\WaterFX_Sample4">
  <WaterFX
    Density="0.98"
    Speed="20.0"
    RefractionFactor="1.0"
    Height="0.05"
    LightColor="1.0, 1.0, 1.0"
    LightPower="40.0"
    LightDirection="-0.9, 0.0"
    InteractionMethod="SOLID"
    SoundEffect=".\Sounds\Water\waterwave.wav"
    Volume="0.6"
    >
      <Image name=".\Images\bg_blue.jpg"/>
  </WaterFX>
</AdTracker>
```

4.19.6 Sample: WaterFX - Thick Gel

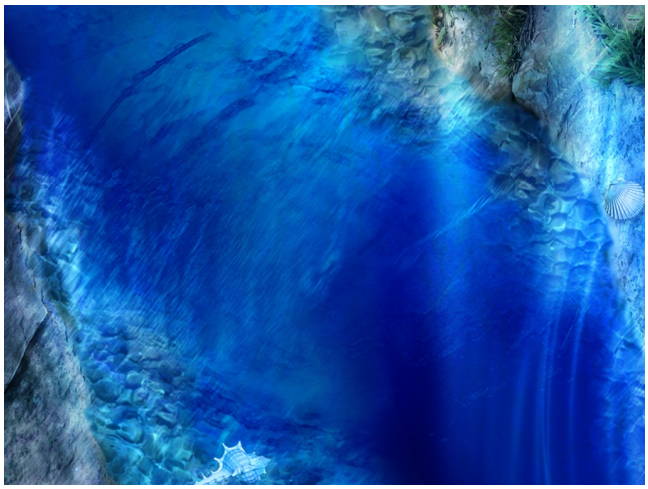


Creates a thick gel effect.

WaterFX_Sample6\root.spark

```
<AdTracker AdName="Manual_Samples\WaterFX_Sample5">
  <WaterFX
    Density="0.99"
    Speed="0.1"
    RefractionFactor="1.0"
    Height="0.05"
    InteractionMethod="solid"
    >
      <Image name=".\Images\bg_blue.jpg"/>
  </WaterFX>
</AdTracker>
```

4.19.7 Sample: WaterFX - Thin Gel



Creates a thin gel effect.

WaterFX_Sample5\root.spark

```
<AdTracker AdName="Manual_Samples\WaterFX_Sample6">
  <WaterFX
    Density="0.92"
    Speed="5.0"
    RefractionFactor="2.0"
    Height="0.05"
    InteractionMethod="Solid"
    >
      <Image name=".\Images\bg_blue.jpg"/>
  </WaterFX>
</AdTracker>
```

4.20 WaterMask

This **Effect** play item is no longer supported. Using the improved WaterFX 120 play item instead is recommended.

This effect is identical to the WaterOverlay 131 effect, except it limits which areas of the presentation are affected. Specifying a **DensityMap** image (64 x 64 RGB, 24 or 32-bit greyscale image) allows you to create variable density over the surface of the water effect. The shade of the **DensityMap** image controls which areas of the presentation will be affected by the **Density** attribute of the **WaterMask**. The more white the color, the less the effect will use the value set in the **Density** attribute (using pure white will result in no water movement at all). The darker the color, the more the effect will use the value set in the **Density** attribute.

You could create a **DensityMap** image with a gradient running from black to white. This would result in the water effect being more fluid on one end while being more dense on the other.

Suggested uses

This effect is good for creating shore lines or other presentations where giving another play item a limited watery look. Could be used for creating a shoreline or beach, where waves only occur in the a desired area of the presentation.

This play item requires at least one child play item.

The **DensityMap** image automatically scales to cover the entire presentation.

4.20.1 Attributes

This play item is no longer supported. Using the improved WaterFX 120 play item instead is recommended.

Density *(Default)*

Controls the fluidity of the ripple effect, including how quickly the waves die down when there is no further motion detection. A lower value makes the effect appear to settle more quickly, whereas a higher value provides a much more liquid appearance. A value of "0.0001" causes no wave effect at all, while a value of "0.9999" makes the effect appear extremely fluid and take longer to settle.

Value Specifications: Type: Decimal; Range: 0.0001 - 0.9999; Default: 0.97

DensityMap *(Default)*

Specifies the path to the density map file. File must be a 64 x 64 RGB, 24 or 32-bit greyscale image.

Density map file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Height *(Default)*

Controls the height of the waves. This along with the **RefractionFactor** attribute value controls how the waves refract. Height also controls wave propagation and how long it takes for them to settle.

Value Specifications: Type: Decimal; Range: 0.001 - 5.0; Default: 0.3

RefractionFactor *(Default)*

The amount of visual refraction that each wave can generate. This attribute controls the look of the waves themselves. The higher the value, the greater the refraction.

Value Specifications: Type: Decimal; Range: 0.001 - 5.0; Default: 0.3

SoundEffect *(Optional)*

Specifies path to a sound file that plays whenever there is motion detection.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Speed *(Default)*

Sets how quickly the waves move. The lowest value is "0.001" which will make for extremely slow movement. The default speed is "0.9" and produces the best results in most cases.

Value Specifications: Type: Decimal; Range: 0.001 - 2.0; Default: 0.9

Volume *(Dependent)*

Sets the volume for the file specified by the optional **SoundEffect** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

4.21 WaterOverlay

This **Effect** play item is no longer supported. Using the improved WaterFX 120 play item instead is recommended.

WaterOverlay is an interactive liquid surface effect that applies a realistic ripple or wave to another play item. Motion detection generates a corresponding set of liquid movements. Attributes control how the ripples or waves look and respond.

4.21.1 Attributes

This play item is no longer supported. Using the improved WaterFX 120 play item instead is recommended.

Density *(Default)*

Controls the fluidity of the ripple effect, including how quickly the waves die down when there is no further motion detection. A lower value makes the effect appear to settle more quickly, whereas a higher value provides a much more liquid appearance. A value of "0.0001" causes no wave effect at all, while a value of "0.9999" makes the effect appear extremely fluid and take longer to settle.

Value Specifications: Type: Decimal; Range: 0.0001 - 0.9999; Default: 0.97

Height (Default)

Controls the height of the waves. This along with the **Refraction** attribute value controls how the waves refract. **Height** also controls wave propagation and how long it takes for them to settle.

Value Specifications: Type: Decimal; Range: 0.001 - 5.0; Default: 0.3

RefractionFactor *(Default)*

The amount of visual refraction that each wave can generate. This attribute controls the look of the waves themselves. The higher the value, the greater the refraction.

Value Specifications: Type: Decimal; Range: 0.001 - 5.0; Default: 0.3

SoundEffect *(Optional)*

Specifies path to a sound file that plays whenever there is motion detection.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Speed *(Default)*

Sets how quickly the waves move. The lowest value is "0.001" which will make for extremely slow movement. The default speed is "0.9" and produces the best results in most cases.

Value Specifications: Type: Decimal; Range: 0.001 - 2.0; Default: 0.9

Volume *(Dependent)*

Sets the volume for the file specified by the optional **SoundEffect** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

4.22 WipeRevealExpand

WipeRevealExpand is a transition **Effect** play item similar to WipeRevealNormal 138, except that when a specified threshold is reached, the revealed area automatically expands to fully reveal the successive child play item.

ScreenXtreme Users! See the special **WipeRevealExpand** samples using the **KeyedCameraViewer** :

WipeRevealExpand with Video 159

WipeRevealExpand Including Video 160

4.22.1 Attributes

AlphaFactor *(Default)*

Controls the transparency setting of the item being wiped away. The higher the setting, the more the item is removed when motion is detected.

Value Specifications: Type: Integer; Range: 1 - 255; Default: 64

CompletionSound *(Optional)*

Sound to play when wiping is complete.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

CompletionThreshold *(Default)*

Controls the percentage of the successive item that needs to be revealed before automatic expansion begins. The lower the setting, the sooner automatic expansion begins. A setting of 1.0 (100%) essentially turns off the expansion attribute since the whole successive item would need to be revealed before it becomes the current one.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 0.8

CompletionVolume *(Dependent)*

Sets the volume for the file specified by the optional **CompletionSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

ExpansionAmount *(Default)*

Controls the amount, in pixels per second, by which the revealed area will expand. The higher the value, the greater the expansion area.

Value Specifications: Type: Integer; Range: 1 - 10000; Default: 500

Increment *(Default)*

Controls the rate, in pixels per second, at which the pixels set by the **ExpansionAmount** attribute change. The higher the value, the faster the expansion area fills up.

Value Specifications: Type: Integer; Range: 1 - 10000 ; Default: 600

Loop *(Default)*

Sets the loop control for the entire **WipeRevealExpand** play item. When set to "True", the last item will reveal the first item in the child play item list and continue through the other items as long as the **WipeRevealExpand** Play item is active. If set to "False", the **WipeRevealExpand** play item will stop with the last child play item listed.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

MaxSoundSamples (*Dependent*)

Controls the maximum number of sounds that will play simultaneously.

Value Specifications: Type: Integer: Range: 1 - 64; Default: 12

MinSoundDelay, **MaxSoundDelay** *(Dependent)*

These two attributes control the delay, in seconds, between the playing of sound effects. Setting different values will create a random delay between the playing of each sound. The minimum value must be equal to or less than the maximum value. Setting the same values for both minimum and maximum will force the sounds to play at regular intervals.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: **MinSoundDelay**: 0.05, **MaxSoundDelay**: 0.1

ThresholdSound *(Optional)*

Sound to play when the threshold setting is reached.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

ThresholdVolume *(Dependent)*

Sets the volume for the file specified by the optional **ThresholdSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

TransitionDelay *(Default)*

Sets the amount of time that interaction will be disabled once the successive play item has been completely revealed. This enables the user to see the revealed content for the time specified before starting to reveal the next item.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 2.0

WipeSoundPath *(Optional)*

Specifies the path to a folder containing one or more sound files that will play (randomly if there is more than one sound file) as wiping occurs.

Files must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

WipeVolume *(Dependent)*

Sets the volume for the files specified by the optional **WipeSoundPath** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

Notes

When this play item is within a SequentialScheduler 39 and the **Loop** attribute is set to "False", the play item will, upon its final transition, terminate, thus allowing the next play item within the SequentialScheduler 39 to start, regardless of the value of the Duration 20 attribute.

For example, if a **WipeRevealExpand** play item inside a SequentialScheduler 39 has its Duration 20 value set to 30 seconds and its **Loop** value set to "False", but it only takes 12 seconds to display all of its child play items, then **WipeRevealExpand** will end after 12 seconds, and the SequentialScheduler 39 will immediately move on to the next play item in the list.

4.22.2 Sample: Basic WipeRevealExpand

Wiping is almost completely transparent. Half of the child play item needs to be revealed before automatic expansion begins, at which time expansion occurs at a fairly quick rate.

WipeRevealExpand1\root.spark

```
<AdTracker AdName="Manual_Samples\WipeRevealExpand1">
  <WipeRevealExpand
    AlphaFactor="96"
    ExpansionAmount="500"
    Increment="500"
    CompletionThreshold="0.50"
    TransitionDelay="2.0"
    Loop="True"
    >
      <Image Name=".\Images\red_curtain.jpg"/>
      <Image Name=".\Images\crowd.jpg"/>
  </WipeRevealExpand>
</AdTracker>
```

4.22.3 Sample: WipeRevealExpand with Sound

Wiping is almost completely transparent. Sounds have been added to the wiping and when the wiping process is completed. 70% of the child play item needs to be revealed before automatic expansion begins, at which time expansion occurs at a fairly quick rate.

WipeRevealExpand2\root.spark

```
<AdTracker AdName="Manual_Samples\WipeRevealExpand2">
  <WipeRevealExpand
    AlphaFactor="64"
    ExpansionAmount="800"
    Increment="800"
    CompletionThreshold="0.7"
    TransitionDelay="2.0"
    Loop="true"
    ThresholdSound=".\Sounds\threshold.wav"
    ThresholdVolume="1.0"
    CompletionSound=".\Sounds\completion.wav"
    CompletionVolume=1.0"
    WipeSoundPath=".\Sounds\wipe"
    WipeVolume="1.0"
    MaxSoundSamples="12"
    MinSoundDelay="0.05"
    MaxSoundDelay="0.1"
    >
      <Image Name=".\Images\red_curtain.jpg"/>
      <Image Name=".\Images\crowd.jpg"/>
  </WipeRevealExpand>
</AdTracker>
```

4.23 WipeRevealNormal

WipeRevealNormal is a transition **Effect** play item that will clear away the contents of the current child play item to reveal a successive child play item. Once the successive item has been completely revealed, it becomes the current item.

4.23.1 Attributes

AlphaFactor *(Default)*

Controls how much of the successive child play item is revealed by motion. The higher the value, the more the item will be revealed.

Value Specifications: Type: Integer; Range: 1 - 255; Default: 64

CompletionSound *(Optional)*

Sound to play when wiping is complete.

File must be Windows®-compliant WAV.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

CompletionThreshold *(Default)*

Controls the percentage of the successive item that needs to be revealed before a complete switch occurs. A setting of "1.0" (100%) means the entire current item must be completely revealed before the successive item becomes the new current item. A setting any lower than "1.0" causes the successive item to become the current item as soon as the set percentage is reached. A very low setting would cause the successive item to become current with very little motion detection.

Once the **CompletionThreshold** is reached, the remainder of the successive item is fully revealed prior to making it the current item.

Value Specifications: Type: Decimal; Range: 0.0 – 1.0; Default: 1.0

CompletionVolume *(Dependent)*

Sets the volume for the file specified by the optional **CompletionSound** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

TransitionDelay *(Default)*

Sets the amount of time that interaction will be disabled once the successive play item has been completely revealed. This enables the user to see the revealed content for the time specified before starting to reveal the next item.

Value Specifications: Type: Decimal; Range: 0.0 and up; Default: 2.0

Loop *(Default)*

Sets the loop control for the entire **WipeRevealNormal** play item. When set to "True", the last item will reveal the first item in the child play item list and continue through the other items as long as the **WipeRevealNormal** play item is active. If set to "False", the **WipeRevealNormal** play item will stop with the last child play item listed.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

Reset *(Default)*

When set to "True" a revealed area will instantly reset once no motion is detected. This is effective for certain effects with which **WipeRevealNormal** may by combined.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

Setting Reset to "True" makes it more difficult to reach the completion threshold, since the revealed area so easily resets.

A low **AlphaFactor** value combined with setting **Reset** to "True" makes it more difficult to reach the specified **CompletionThreshold**, since the lower the transparency, the more motion detection is needed to reach the threshold.

Notes

When this play item is within a SequentialScheduler 39 and the **Loop** attribute is set to "False", the play item will, upon its final transition, terminate, thus allowing the next play item within the SequentialScheduler 39 to start, regardless of the value of the Duration 20 attribute.

For example, if a **WipeRevealNormal** play item inside a SequentialScheduler 39 has its Duration 20 value set to 30 seconds and its **Loop** value set to "False", but it only takes 12 seconds to display all of its child play items, then **WipeRevealNormal** will end, and the SequentialScheduler 39 will immediately move on to the next play item in the list.

4.23.2 Sample: Basic Wipe Effect

Creates a basic repeating wipe effect of two child play items. Wiping is completely transparent. 70% the current item needs to be revealed before the successive item becomes current at which point the effect will be inactive for 2 seconds. A sound will play at full volume when wipe is complete.

WipeRevealNormal\root.spark

```
<AdTracker AdName="Manual_Samples\WipeRevealNormal">
  <WipeRevealNormal
    AlphaFactor="255"
    CompletionThreshold="0.7"
    TransitionDelay="2.0"
    Loop="true"
    CompletionSound=".\Sounds\completion.wav"
    CompletionVolume="1.0"
    >
      <Image Name=".\Images\metal.jpg"/>
      <Image Name=".\Images\marble.jpg"/>
  </WipeRevealNormal>
</AdTracker>
```

4.24 WipeRevealReset

WipeRevealReset is an **Effect** play item that works with two child play items with the first item completely covering the second. Upon motion detection, the second item is revealed in the area of motion detection only. When motion is no longer detected, the affected area closes up and is covered by the first item again.

ScreenXtreme Users! See the special **WipeRevealReset** sample using the **KeyedCameraViewer**:

WipeRevealReset Including Video 161

4.24.1 Attributes

The overall reset effect is controlled by the combination of the **ContractionAmount** attribute and the **Decrement** attribute. For best results it is recommended to set both of these attributes to at least 100.

AlphaFactor *(Default)*

Controls how much of the secondary play item is revealed per motion detection. The higher the setting, the more the item is revealed upon motion.

Value Specifications: Type: Integer; Range: 1 - 255; Default: 64

ContractionAmount *(Default)*

Controls how quickly, in pixels per second, the revealed area contracts (closes up over the secondary item).

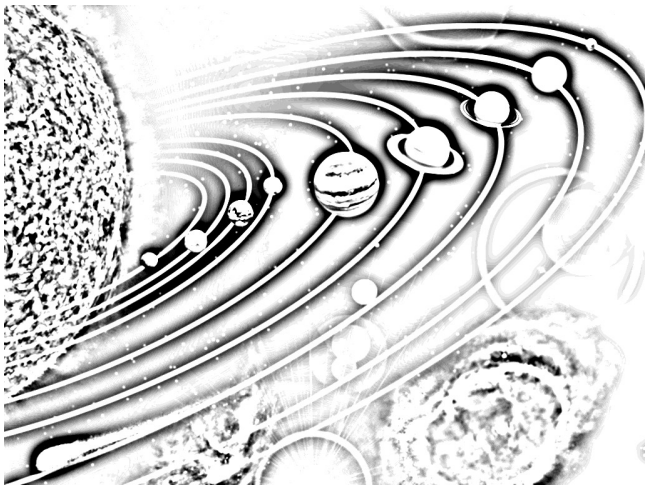
Value Specifications: Type: Integer; Range: 1 - 1000; Default: 600

Decrement *(Default)*

Controls how quickly, in pixels per second, the area set by the contraction area resets to the primary play item. The larger the value, the faster the contraction area will contract.

Value Specifications: Type: Integer; Range: 1 - 1000; Default: 500

4.24.2 Sample: Basic WipeRevealReset Effect



A basic **WipeRevealReset** effect that resets at a fairly quick rate.

WipeRevealReset.spark

```
<AdTracker AdName="Manual_Samples\WipeRevealReset">
  <WipeRevealReset
    AlphaFactor="96"
    ContractionAmount="450"
    Decrement="300"
    >
      <Image Name=".\Images\planet_coordinates_2.png"/>
      <Image Name=".\Images\planet_coordinates_1.png"/>
  </WipeRevealReset>
</AdTracker>
```

5 Play Items - Players

Player play items display or run one or more media files, including images, videos, Flash®, and sound.

Directory 144
File 147
FlashFile 149
Image 150
KeyedCameraViewer 151 (ScreenXtreme only)
MusicPlayer 162
Video 164

5.1 Directory

This is a **Player** play item that displays multiple media files in sequence.

Suggested uses

Can be used instead of listing individual child play items when a sequence of media files is desired.

This play item must be the child item of a play item that allows for sequencing of media files. Play items that work with **Directory** include, AlphaTransition 49, BasicTiles 51, WipeRevealExpand 133, and WipeRevealNormal 138. For **Directory** to run on its own (simply displaying multiple media files) or as a child of a play item that normally does not sequence child items, it should be placed in a SequentialScheduler 39.

5.1.1 Attributes

Filter *(Default)*

Allows for file filtering using common wildcards "*" and "?". Multiple filters may be specified, separated by commas (e.g. *.jpg, g*.*, image*.tga).

Value Specifications: Type: Enumerated; Default: "*.*"

Name *(Required)*

Specifies the path to the folder (and optional subfolders) that contain the desired files.

Notes

1. Sequencing of files works according to standard alphanumeric sorting conventions (also applies to multiple folders when the **Recursive** attribute is set to "True").

Example

The following files would be sorted as follows:

- file01.img
- file10.img
- file11.img
- file2.img
- file4.img

2. Supported file formats are:

- AVI (see Note 3)
- BMP
- DDS
- DIB
- HDR
- JPG (See Note 4)
- MPEG (See Note 5)
- PFM
- PNG
- PPM
- SWF (See Note 6)
- TGA

3. Viewing AVI files requires that the CODEC (compression/decompression software) associated with their creation be installed on the system in which they are running. Some video player programs, such as Windows Media® Player, include workarounds that allow AVI files to play without actually installing the associated CODEC. This means that while you may be able to view particular video files in Windows Media® Player, you may not be able to run them in Dazzler.

4. If Dazzler is having difficulties displaying a particular JPG file, try saving it in a more common JPG format or in another supported format.

5. Running MPEG2 and MPEG4 files may necessitate acquiring a third-party CODEC, such as those available from Elecard Ltd. Disclaimer: GestureTek nor its Resellers warrant third-party products and services unless explicitly stated otherwise.

6. Flash® files must be registered using the GestureTek Flash Register security application. The GestureTek Flash SDK outlines a technique for adding interactions between the user and Flash® objects. For more information, see the GestureTek Flash SDK documentation or contact your GestureTek reseller.

7. Video file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Recursive *(Default)*

Allows for the sequencing of files in all subfolders of the folder specified in the **Name** attribute.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

5.1.2 Sample: Directory Play Item with SequentialScheduler

An example of using the **Directory** play item within a **SequentialScheduler** play item. Media files located in the "images" folder will display for five seconds each. Subfolders will be ignored. Only files ending with ".jpg" displayed.

The images directory being used with this sample contains only jpg files. To demonstrate the filter, you may copy in other types of supported image files to the image folder.

Image file names must be comprised of letters, numbers, underscores and dashes only.

DirectoryWithScheduler\root.spark

```
<AdTracker AdName="Manual_Samples\DirectoryWithScheduler">
  <SequentialScheduler>
    <Directory Duration="5"
      Name=".\Images"
      Filter="*.jpg"
      />
  </SequentialScheduler>
</AdTracker>
```

5.1.3 Sample: Directory Play Item with BasicTiles

An example of using the **Directory** play item within a **BasicTiles** play item. The **BasicTiles** effect is sequencing through all the files in the "images" folder, ending with ".jpg".

The images directory being used with this sample contains only jpg files, To demonstrate the filter, you may copy in other types of supported image files to the image folder.

Image file names must be comprised of letters, numbers, underscores and dashes only.

DirectoryWithTiles\root.spark

```
<AdTracker AdName="Manual_Samples\DirectoryWithTiles">
  <BasicTiles
    HorizontalTiles="1"
    VerticalTiles="1"
    RotationSpeed="90.0"
    Direction="Y"
    BackgroundColor="1.0,0.0,0.0"
    Perspective="true"
    CompletionThreshold="0.8"
    TransitionDelay="2.0"
    Loop="True"
    >
      <Directory Name=".\Images" filter="*.jpg" />
  </BasicTiles>
</AdTracker>
```

5.2 File

This is a **Player** play item that runs any supported image, video, or animation file.

5.2.1 Attributes

Name *(Required)*

Specifies the file to display. Supported file formats are:

- AVI (See Note 3)
- BMP
- DDS
- DIB
- HDR
- JPG (See Note 4)
- MOV
- MPEG (See Note 5)
- PFM
- PNG
- PPM
- SWF (See Note 6)
- TGA

Notes

1. The **File** play item works very similar to the FlashFile 149, Image 150, and Video 164 play items. While these other play items only work with their particular file type, the **File** play item is able to work with any supported file type.

2. Files will be automatically scaled to the full screen resolution dimensions set in the **dazzler.cfg** file. For best results, files should be created using those same resolution dimensions.

3. Viewing AVI files requires that the CODEC (compression/decompression software) associated with their creation be installed on the system in which they are running. Some video player programs, such as Windows Media® Player, include workarounds that allow AVI files to play without actually installing the associated CODEC. This means that while you may be able to view particular video files in Windows Media® Player, you may not be able to run them in Dazzler.

4. If Dazzler is having difficulties displaying a particular JPG file, try saving it in a more common JPG format or in another supported format.

5. Running MPEG2 and MPEG4 files may necessitate acquiring a third-party CODEC, such as those available from Elecard Ltd. **Disclaimer:** Neither GestureTek nor its Resellers warrant third-party products and services unless explicitly stated otherwise.

6. The playing of MOV (QuickTime) files requires that the QuickTime Player be installed (version 7.6).

7. Flash® files must be registered using the GestureTek Flash Register security application. The GestureTek Flash SDK outlines a technique for adding interactions between the user and Flash® objects. For more information, see the GestureTek Flash SDK documentation or contact your GestureTek reseller.

8. If the **File** play item is placed within a **Scheduler**, the Duration attribute setting will have precedence over the length of the Flash® file. In other words if a specified Flash® file is 55 seconds in length and the Duration attribute is set to 30 seconds, the **File** play item will stop after 30 seconds.

9. File names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

5.2.2 Sample: Display Media File

Displays a media file, which in this case is an image.

File\root.spark

```
<AdTracker AdName="Manual_Samples\File">
  <File Name=".\Images\crowd.jpg"/>
</AdTracker>
```

5.3 FlashFile

This is a **Player** play item that will run a Flash® (SWF) file.

Notes

1. File will be displayed full screen, regardless of actual dimensions.

2. For more information on using Flash® files with Dazzler, see the GestureTek Flash SDK documentation or contact your GestureTek reseller.

5.3.1 Attributes

Name *(Required)*

Specifies the Flash® (SWF) file to play.

Notes

1. Flash® files must be registered using the GestureTek Flash Register security application. The GestureTek Flash SDK outlines a technique for adding interactions between the user and Flash® objects. For more information, see the GestureTek Flash SDK documentation or contact your GestureTek reseller.

2. If the **FlashFile** play item is placed within a **Scheduler**, the Duration 20 attribute setting will have precedence over the length of the Flash® file. In other words if a specified Flash® file is 55 seconds in length and the Duration 20 attribute is set to 30 seconds, the **FlashFile** play item will stop after 30 seconds.

3. File names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

5.3.2 Sample: Display Flash File

Plays a Flash® file.

FlashFile\root.spark

```
<AdTracker AdName="Manual_Samples\FlashFile">
  <FlashFile Name=".\Swf\welcome.swf"/>
</AdTracker>
```

5.4 Image

This is a **Player** play item that displays a single image file.

The image will be displayed full screen, regardless of actual dimensions.

5.4.1 Attributes

Name (*Required*)

Specifies the image file to display. Supported image file formats are:

- BMP
- DDS
- DIB
- HDR
- JPG (See Note 1)
- PFM
- PNG
- PPM
- TGA

Notes

1. If Dazzler is having difficulties displaying a particular JPG file, try saving it in a more common JPG format or in another supported format.
2. Images will be automatically scaled to the full screen resolution dimensions set in the **dazzler.cfg** file. For best results, images should be created using those same resolution dimensions.
3. File names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

5.4.2 Sample: Display Image File

Displays an image file.

Image\root.spark

```
<AdTracker AdName="Manual_Samples\Image">
  <Image Name=".\Images\hawaii.jpg"/>
</AdTracker>
```

5.5 KeyedCameraViewer

Important! *This play item is only available with ScreenXtreme.*

The **KeyedCameraViewer** displays the current real-time tracked camera image. The result is that users see their video image interacting with other play items. The **KeyedCameraViewer** can be placed as either the parent or child play item, depending on the type of results desired.

Implementing the KeyedCameraViewer

There are two basic implementations of the **KeyedCameraViewer**:

1. Providing a transparent video image over other play items. This displays a video image upon motion detection. When motion is detected the video image is displayed and then fades out based on the **Duration** 20 setting (ScreenXtreme Tracker Advanced settings, Mask tab). The **KeyedCameraViewer** can be the first play item, with all other play items nested within it. This means the video image will always be overlaid on top of the entire presentation. Alternatively it can be nested somewhere within the overall presentation. This allows other effects to be applied to the video image and/or for the video image to appear only at some point of the overall presentation.

2. As a permanent background. The **KeyedCameraViewer** can act as a video background for other play items. Setting the **AlphaFactor** attribute to "False" renders the video image opaque (no transparency).

Creating a transparent permanent video display

There is a special implementation of the **KeyedCameraViewer** that allows for the displaying of a permanent transparent video image. Normally when wanting a transparent video image, the image will only appear upon motion. Where it is preferable to display a transparent video image whether motion is detected or not, the **KeyedCameraViewer** with its transparency turned off can be placed inside the **AlphaOverlay** shader filter, thus allowing for the display of a permanent transparent video image.

5.5.1 Attributes

AlphaEnabled *(Default)*

When set to "True" (default setting), alpha blending is activated, thus allowing the camera image to be keyed. Only those items currently moving will be displayed. When set to "False", no alpha blending will be applied. No child items will be displayed.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

AlphaFactor *(Dependant)*

If **AlphaEnabled** is set to "True", this value can be used to reduce the alpha of the entire camera image. A setting of "0.0" will render the camera image completely transparent (invisible), while "1.0" (100%) doesn't reduce the alpha at all.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

FlipHorizontal *(Default)*

When set to "True", the video image as specified in the ScreenXtreme Tracker will be flipped horizontally.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

FlipVertical *(Default)*

When set to "True", the video image as specified in the ScreenXtreme Tracker will be flipped vertically.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

Position *(Default)*

Specifies the top-left corner position of the video image. This attribute is ignored if the **Size** attribute is set to "FullScreen".

Value Specifications: Type: Integer; Range: 0 and up ; Default: "0,0"

Size *(Default)*

Specifies the size of the camera image to be displayed. Instead of an integer value, the default setting, "FullScreen" may be used.

Value Specifications: Type: Integer/Enumerated; Range: 1 and up (or "FullScreen"); Default: "FullScreen"

Width and height should be specified as an integer pair (except when using "FullScreen"). For example, a size of 500 pixels wide by 300 high would be as follows: Size="500,300"

Size only affects the display of the video image, not the tracking, which always remains full screen.

5.5.2 Sample: ParticlesInteractAnimate with Video

KeyedCameraViewer sample with animated particles over a video image, which in turn is over an image.

KCV-ParticlesInteractAnimate\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-ParticlesInteractAnimate">
  <ParticlesInteractAnimate
    ParticleCount="6"
    ParticleWidth="96"
    ParticleHeight="96"
    ParticleRadius="30"
    ParticleTexturePath=".\Images\balls"
    KineticFriction="2.0"
    MaxForce="10"
    EnforceBorders="true"
    BlendMode="normal"
    EnableRotation="true"
    RotationSpeed="0.0"
    Collisions="true"
    AnimationCycleTime="1.0"
    AnimationMode="LOOP"
    RandomStartFrame="True"
    >
      <KeyedCameraViewer
        FlipHorizontal="False"
        Position="0,0"
        Size="FullScreen"
        AlphaEnabled="True"
        >
          <Image Name=".\Images\logo.jpg"/>
      </KeyedCameraViewer>
  </ParticlesInteractAnimate>
</AdTracker>
```

5.5.3 Sample: Bubbles in Water with Video

A complex **KeyedCameraViewer** sample using **ParticlesEmitMotion**. The particles emit over a **WaterFX** play item which is in turn over a transparent, permanent video image over an image.

KCV-BubblyWater\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-BubblyWater">
  <ParticlesEmitMotion
    EmitRate="200"
    MinWidth="12"
    MaxWidth="25"
    MaintainAspectRatio="True"
    MinLiveTime="1.0"
    MaxLiveTime="1.5"
    FadeTime="1.0"
    Velocity="0.0, -50.0"
    Acceleration="0.0, -150.0"
    ExpandRate="1.0"
    ParticleTexturePath=".\Images\bubble.png"
    BlendMode="Normal"
    >
      <WaterFX
        Density="0.98"
        Speed="20.0"
        RefractionFactor="0.1"
        Height="0.05"
        InteractionMethod="Solid"
        >
          <ChildPlayer>
            <Image Name=".\Images\screen_bg.png"/>
            <ShaderFilter Name="VisualFilters\AlphaOverlay.pso" AlphaEnabled="True" g_
              <KeyedCameraViewer
                FlipHorizontal="False"
                Position="0,0"
                Size="FullScreen"
                AlphaEnabled="False"
                />
            </ShaderFilter>
          </ChildPlayer>
      </WaterFX>
  </ParticlesEmitMotion>
</AdTracker>
```

5.5.4 Sample: IceCrack with Sound and Video

An **IceCrack** sample with the **KeyedCameraViewer** applied.

KCV-IceCrack\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-IceCrack">
  <KeyedCameraViewer
    FlipHorizontal="False"
    Position="0,0"
    Size="FullScreen"
    AlphaEnabled="True"
    >
    <IceCrack
      ComplexityLevel="7"
      SplitMin="0.2"
      SplitMax="0.8"
      CrackForce="2.0"
      CrackRadius="128"
      FadeThreshold="0.8"
      ShrinkSpeed="5.0"
      FadeSpeed="2.0"
      Loop="True"
      SoundPath=".\sounds\ice"
      Volume="0.5"
      MaxSounds="12"
      MinSoundDelay="0.05"
      MaxSoundDelay="0.1"
      >
        <Image Name=".\Images\icetop.jpg"/>
        <Image Name=".\Images\iceshark.jpg"/>
    </IceCrack>
  </KeyedCameraViewer>
</AdTracker>
```

5.5.5 Sample: ParticlesReset with Video

KeyedCameraViewer sample using **ParticlesReset**. Particles are over a video image, which is over an image.

KCV-ParticlesReset\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-ParticlesReset">
  <ParticlesReset
    GridWidth="16"
    GridHeight="12"
    ParticleRadius="0.0"
    ParticleTexturePath=".\Images\ParticlesResetTiles"
    InteractionForce="10"
    KineticFriction="1.0"
    ResetForce="2.0"
    MotionBlur="0.8"
    TopLeft="0.0,0.0"
    BottomRight="0.0,0.0"
    >
      <KeyedCameraViewer
        FlipHorizontal="False"
        Position="0,0"
        Size="FullScreen"
        AlphaEnabled="True"
        >
          <Image Name=".\Images\earth.jpg" />
      </KeyedCameraViewer>
  </ParticlesReset>
</AdTracker>
```

5.5.6 Sample: Popcorn with Video

KeyedCameraViewer sample using **ParticleInteractAnimate**. Animated popcorn appears over permanent video image.

KCV-Popcorn\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-Popcorn">
  <ParticlesInteractAnimate
    ParticleTexturePath=".\Images\popcorn"
    ParticleCount="850"
    ParticleWidth="64"
    ParticleHeight="64"
    ParticleRadius="40"
    MaxForce="10.0"
    KineticFriction="2.0"
    RandomStartAngle="True"
    EnableRotation="True"
    RotationSpeed="0.0"
    RandomStartFrame="False"
    AnimationCycleTime="0.5"
    AnimationMode="TWOSHOTDELAY"
    AnimationMinDelay="6.0"
    AnimationMaxDelay="10.0"
    BlendMode="Normal"
    Reset="True"
    ResetForce="5"
    EnforceBorders="False"
    Collisions="False"
    >
      <KeyedCameraViewer
        FlipHorizontal="False"
        Position="0,0"
        Size="FullScreen"
        AlphaEnabled="False"
        />
  </ParticlesInteractAnimate>
</AdTracker>
```

5.5.7 Sample: Venetian Blinds with Video

KeyedCameraViewer sample using **BasicTiles**. Video image appears over vertical strips

KCV-VenetianBlinds\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-VenetianBlinds">
  <KeyedCameraViewer
    FlipHorizontal="False"
    Position="0,0"
    Size="FullScreen"
    AlphaEnabled="True"
    >
      <BasicTiles
        HorizontalTiles="10"
        VerticalTiles="1"
        RotationSpeed="270.0"
        Direction="Y"
        BackgroundColor="0.5,0.5,0.5"
        Perspective="True"
        CompletionThreshold="0.8"
        TransitionDelay="2.0"
        Loop="True"
        >
         <Image Name=".\Images\marble.jpg"/>
          <Image Name=".\Images\metal.jpg"/>
      </BasicTiles>
  </KeyedCameraViewer>
</AdTracker>
```

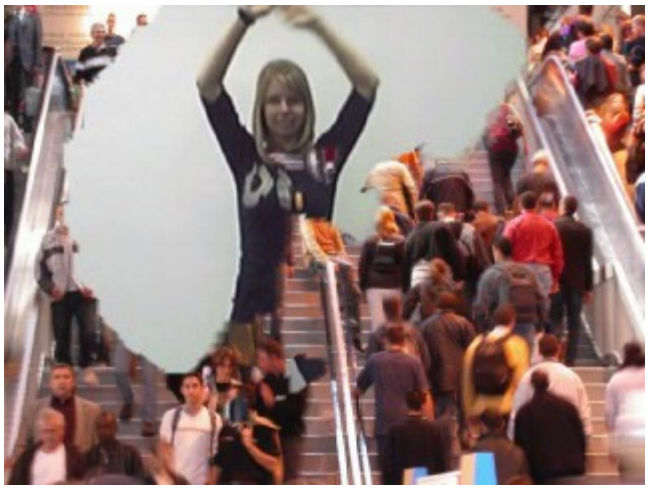
5.5.8 Sample: WipeRevealExpand with Video

One of two **KeyedCameraViewer** samples using **WipeRevealExpand**. The video image is overlaid on top of the **WipeRevealExpand** effect.

KCV-WipeRevealExpand1\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-WipeRevealExpand1">
  <KeyedCameraViewer
    FlipHorizontal="False"
    Position="0,0"
    Size="FullScreen"
    AlphaEnabled="True"
    >
      <WipeRevealExpand
        AlphaFactor="96"
        ExpansionAmount="500"
        Increment="500"
        CompletionThreshold="0.50"
        TransitionDelay="2.0"
        Loop="True"
        >
          <Image Name=".\Images\metal.jpg"/>
          <Image Name=".\Images\marble.jpg"/>
      </WipeRevealExpand>
  </KeyedCameraViewer>
</AdTracker>
```

5.5.9 Sample: WipeRevealExpand Including Video



Second of two **KeyedCameraViewer** samples using **WipeRevealExpand**. Here the video image is one of three items that are revealed by wiping.

KCV-WipeRevealExpand2\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-WipeRevealExpand2">
  <WipeRevealExpand
    AlphaFactor="96"
    ExpansionAmount="500"
    Increment="500"
    CompletionThreshold="0.50"
    TransitionDelay="2.0"
    Loop="False"
    >
      <Image Name=".\Images\metal.jpg"/>
      <Image Name=".\Images\marble.jpg"/>
      <KeyedCameraViewer
        FlipHorizontal="False"
        Position="0,0"
        Size="FullScreen"
        AlphaEnabled="False"
        />
  </WipeRevealExpand>
</AdTracker>
```

5.5.10 Sample: WipeRevealReset Including Video

KeyedCameraViewer sample using **WipeRevealReset**. The video image appears to wipe away the first image in order to reveal the second image. When motion stops, the first image covers the second again.

KCV-WipeRevealReset\root.spark

```
<AdTracker AdName="Manual_Samples\KCV-WipeRevealReset">
  <KeyedCameraViewer
    FlipHorizontal="False"
    Position="0,0"
    Size="FullScreen"
    AlphaEnabled="True"
    >
      <WipeRevealReset
      AlphaFactor="96"
      ContractionAmount="650"
      Decrement="500"
      >
        <Image Name=".\Images\bmw_lineart.jpg"/>
        <Image Name=".\Images\bmw_colour.jpg"/>
      </WipeRevealReset>
  </KeyedCameraViewer>
</AdTracker>
```

5.6 MusicPlayer

This is a general-purpose sound player, most suitable for background music or continuous sound effect. Sound plays as soon as the play item runs.

Notes

1. In order to use **MusicPlayer** to provide background music for another play item, the other play item must be the child play item to the **MusicPlayer**.
2. To play more than one sound file in sequence, multiple instances of **MusicPlayer** should be placed within a **SequentialScheduler** 39 play item.
3. To supply background music to a scheduler itself, rather than having the scheduler run through a sequence of sound files, the scheduler must be the child play item of the **MusicPlayer** as explained in Note 1 above.
4. If the **MusicPlayer** play item is placed within a scheduler play item, the **Duration** 20 attribute setting will have precedence over the length of the sound file. In other words if a specified sound file is 55 seconds in length and the **Duration** 20 attribute is set to 30 seconds, the **MusicPlayer** play item will stop after 30 seconds.
5. If the **MusicPlayer** play item is placed within a scheduler play item and the **Duration** 20 attribute is omitted, then the duration will be automatically set to the length of the sound file.

5.6.1 Attributes

Loop *(Default)*

When set to "True", the sound file will repeat indefinitely. If "False", the sound file will play through only once.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "True"

For seamless loops, wav files are recommended, since MP3 files may give unexpected results.

SoundFile *(Required)*

Specifies the path to the desired sound file. The file may be a Windows®-compliant WAV or an MP3.

Sound file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

Volume *(Default)*

Sets the playback volume for the file specified by the required **SoundFile** attribute.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: 1.0

5.6.2 Sample: Simple MusicPlayer

Provides background music to an **Image** play item. Loops indefinitely.

MusicPlayer\root.spark

```
<AdTracker AdName="Manual_Samples\MusicPlayer">
  <MusicPlayer
    SoundFile=".\Sounds\music.wav"
    Loop="True"
    Volume="1.0"
    >
  <Image Name=".\Images\earth.jpg"/>
  </MusicPlayer>
</AdTracker>
```

5.7 Video

This is a **Player** play item that will run a video file.

5.7.1 Attributes

BackGroundClr *(Default)*

An RGB value that specifies the color for the screen background when **FullScreen** is set to "False". For more information about setting color attributes, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: "1.0,1.0,1.0" (white)

FullScreen *(Default)*

If set to "True", video will be displayed full screen. Dimensions will be scaled to fit the full resolution set in the Dazzler configuration file (**dazzler.cfg**).

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

Loop *(Default)*

When set to "False", video will play one time only. When set to "True", video will repeat continuously.

Value Specifications: Type: Boolean; Range: "True" or "False"; Default: "False"

Name *(Required)*

Specifies the video file to play. Video file formats supported are AVI, MPEG and MOV (see notes below).

Notes

1. Viewing AVI files requires that the CODEC (compression/decompression software) associated with their creation be installed on the system in which they are running. Some video player programs, such as Windows Media® Player, include workarounds that allow AVI files to play without actually installing the associated CODEC. This means that while you may be able to view particular video files in Windows Media® Player, you may not be able to run them in Dazzler.

2. Running MPEG2 and MPEG4 files may necessitate acquiring a third-party CODEC, such as those available from Elecard Ltd. Disclaimer: Neither GestureTek nor its Resellers warrant third-party products and services unless explicitly stated otherwise.

3. The playing of MOV (QuickTime) files requires that the QuickTime Player be installed (version 7.6).

4. If the Video play item is placed within a Scheduler, the Duration 20 attribute setting will have precedence over the length of the video. For example, if a specified video file is 55 seconds in length and the Duration 20 attribute is set to 30 seconds, the Video play item will stop after 30 seconds.

5. Video file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: FilePath

5.7.2 Sample: Play Video File

Plays a video file according to default values.

Video1\root.spark

```
<AdTracker AdName="Manual_Samples\Video1">
  <Video
    Name=".\Videos\dazzler_logo_anim_640x480.mpg"
    FullScreen="False "
    Loop="False"
    BackGroundClr="1.0,1.0,1.0"
    />
</AdTracker>
```

5.7.3 Sample: Play Video File, Full Screen

Continuously plays a video file in **FullScreen** mode.

Video2\root.spark

```
<AdTracker AdName="Manual_Samples\Video2">
  <Video
    Name=".\Videos\dazzler_logo_anim_640x480.mpg"
    FullScreen="True"
    Loop="True"
    />
</AdTracker>
```

6 Play Items - Utilities

Utility play items are tools to help developers create effective Spark files.

AdTracker 167
FrameRate 169
ScreenShot 170
Tracker 172

6.1 AdTracker

This is a **Utility** play item for tracking interactions with specific spark files.

Notes

1. The amount of user interaction with the specified Spark file is measured by the average number of polygons detected by the tracker during the play time of the Spark file. The information is collected in the **AdTracker.log** file located in the **Logs** subdirectory. The default location is **C:\GestureTek\ProductName\Logs**.

2. Once the log grows to a certain size (specified by **MaxHours** in the **MiscEffects.cfg** file), it will keep the last portion of the file (specified by **MinHours**) and remove the preceding data. The **MiscEffects.cfg** file is located in the **Config** subdirectory. The default location is **C:\GestureTek\ProductName\Config**.

6.1.1 Attributes

AdName*(Required)*

Specifies the name of the Spark file to track.

Spark file names must be comprised of letters, numbers, underscores and dashes only.

Value Specifications: Type: String

6.1.2 Sample: Track Ad

Tracks the interactions of a spark file named "Bubbles".

```
<?xml version="1.0"?>
<AdTracker AdName="Demos\Combos\Bubbles">
  <ParticlesEmitMotion
   EmitRate="100"
   MinWidth="16"
   MaxWidth="32"
   MinHeight="16"
   MaxHeight="16"
   MaintainAspectRatio="true"
   MinLiveTime="0.5"
   MaxLiveTime="2.0"
   FadeTime="0.5"
   Velocity="0.0, -50.0"
   Acceleration="0.0, -150.0"
   ExpandRate="1.0"
   ParticleTexturePath=".\Images\bubble.tga"
   BlendMode="normal"
   SoundEffectPath=".\Sounds\bubble2.wav"
   Volume="1.0"
   MaxSoundSamples="10"
   MinSoundDelay="0.05"
   MaxSoundDelay="0.05"
   >
   <WaterFX
     Density="0.50"
     Speed="20"
     RefractionFactor="1.0"
     Height="0.01"
     LightColor="0.8, 0.8, 0.8"
     LightPower="15.0"
     LightDirection="0.65, 0.0"
     >
     <Image Name=".\Images\Belugas.jpg"/>
   </WaterFX>
  </ParticlesEmitMotion>
</AdTracker>
```

6.2 FrameRate

This is a **Utility** play item that displays Dazzler's current rendered frame rate. Useful for performance testing.

Notes

1. **FrameRate** can be activated any time Dazzler is running by pressing **Ctrl-F**. Changing the color of the displayed frame rate may be done by setting the **Sparky**>**FusePlayer FrameRateColour** attribute of the **Sparky.cfg** file, which is located in your **Config** folder. Restart Dazzler to apply changes.

2. If a **FrameRate** play item is currently running from a Spark file, pressing **Ctrl-F** will create undesirable results.

3. For **FrameRate** to display correctly when used in a Spark file, it must be placed as the last item within another play item that allows for the combining of play items, such as the ChildPlayer 33.

6.2.1 Attributes

Color *(Default)*

An RGB value that sets the color of the frame rate display text. For more information about setting color attributes, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: "1.0, 1.0, 1.0"

The color chosen should be one that is in good contrast to the current display.

RefreshDelay *(Default)*

Specifies the number of seconds between frame rate updates.

Value Specifications: Type: Decimal; Range: 0.1 and up; Default: 1.0

6.2.2 Sample: Display Frame Rate Yellow

Displays current frame rate in yellow over an **Image** play item. Frame rate refreshes every two seconds.

FrameRate\root.spark

```
<AdTracker AdName="Manual_Samples\FrameRate">
  <ChildPlayer>
    <Image Name=".\Images\moon.jpg" />
    <FrameRate RefreshDelay="2.0" Color="1.0, 1.0, 0.0" />
  </ChildPlayer>
</AdTracker>
```

6.3 ScreenShot

This is a **Utility** play item that saves the display of the immediately nested play item to a jpg file.

The Dazzler display will pause shortly while the screenshot is saved.

6.3.1 Attributes

DelayTime *(Default)*

Specifies the time, in seconds, before the screenshot will be taken. The default value of "0.00" means that the screenshot will be taken as soon as the nested play item starts.

Value Specifications: Type: Decimal; Range: 0.00 and up; Default: 0.00

Name *(Required)*

Specifies the file path of the image file to be saved.

Value Specifications: Type: FilePath (with a jpg extension)

Notes

1. All files must be saved to a folder within the **Resources** folder. If no path is specified, but a file name is, the file will automatically be saved to the **Resources** folder. If the **Name** field is left blank, the screenshot will not be saved. For more information, see Understanding File Paths 26.

2. Only saving to existing folders is supported.

3. All image file names must be comprised of letters, numbers, underscores and dashes only.

4. While Dazzler will allow you to use a file extension other than "jpg", specifying "jpg" is recommended.

Files with the same name will be overwritten without warning.

6.3.2 Sample: ScreenShot

Takes a screenshot of a **WaterFX** sample after five seconds and saves it to a file, named **`GFXscreenShot.jpg`** stored in the **Resources** folder.

screenshot\root.spark

```
<AdTracker AdName="Manual_Samples\ScreenShot">
  <ScreenShot
    Name="GFXscreenShot.jpg"
    DelayTime="5.0"
    >
      <WaterFX
        Density="0.98"
        Speed="20.0"
        RefractionFactor="1.0"
        Height="0.05"
        InteractionMethod="Solid"
        >
          <Image Name=".\Images\bg_blue.jpg"/>
      </WaterFX>
  </ScreenShot>
</AdTracker>
```

6.4 Tracker

This is a **Utility** play item that displays a tracking data outline, allowing you to see the outline of motion detection within the presentation area. Useful for testing purposes.

Notes

1. Alternatively, **Tracker** can be activated any time Dazzler is running by pressing **Ctrl-T**. Changing the color of the tracker outline may be done by setting the **Sparky**>**FusePlayer TrackerColour** attribute of the **Sparky.cfg** file, which is located in your **Config** folder.

2. If a **Tracker** play item is currently running from a Spark file, pressing **Ctrl-T** will create undesirable results.

3. For **Tracker** to display correctly when used in a Spark file, it must be placed as the last item within another play item that allows for the combining of play items, such as the ChildPlayer 33.

6.4.1 Attributes

Color *(Default)*

An RGB value that sets the color of the displayed tracking data. For more information about setting color attributes, see Setting Color Values 19.

Value Specifications: Type: Decimal; Range: 0.0 - 1.0; Default: "1.0, 1.0, 1.0"

The color chosen should be one that is in good contrast to the current display.

6.4.2 Sample: Display Tracker

Displays tracking data in black over **ParticlesInteractAnimate**.

Tracker\root.spark

```
<AdTracker AdName="Manual_Samples\Tracker">
  <ChildPlayer>
    <Include Name=".\ParticlesInteractAnimate\root.spark" />
    <Tracker Color="0.0, 0.0, 0.0" />
  </ChildPlayer>
</AdTracker>
```

7 Dazzler Mouse Tracker

The Dazzler Mouse Tracker...

- Allows you to use GestureTek's GestPoint and ClearTouch technologies to run Dazzler.
- Allows you to use Dazzler without a camera, making it an excellent tool for development and testing.

ScreenXtreme Users: To view camera image, the ScreenXtreme Tracker is required.

7.1 Opening the Console

Open the Dazzler Mouse Tracker Console by running **DazzlerMouse.exe** from the Tracker folder. The Dazzler Mouse console will appear.

7.2 Running the Mouse Tracker

To start the Mouse Tracker, click the **START** button or wait for it to auto-connect. (If the Mouse Tracker auto-connects, it immediately minimizes to the Windows system tray).

7.3 Running Dazzler using Dazzler Mouse

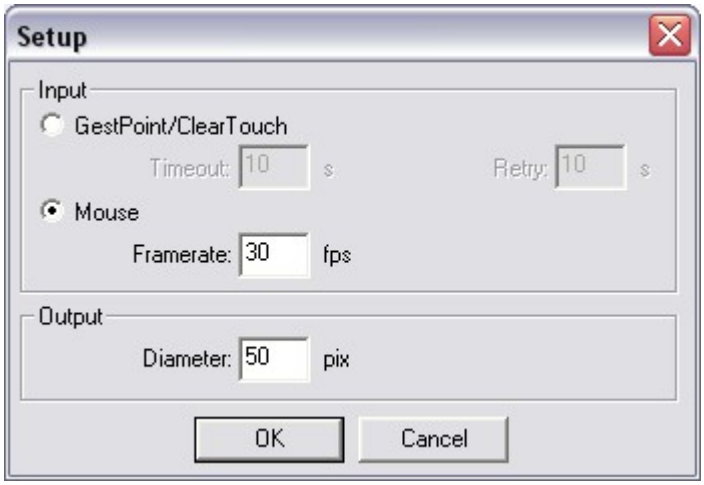
Once the Dazzler Mouse Tracker is running, you can start Dazzler from the Dazzler Mouse console by clicking on **START DAZZLER**.

7.4 Stopping the Mouse Tracker

When the Tracker is running, the **START** button changes to **STOP**. Click the **STOP** button to stop the Tracker.

7.5 Setup

To access the Dazzler Mouse Tracker settings, make sure the Mouse Tracker is not running, and click the **Setup** button. The following window will appear:



Input

GestPoint/ClearTouch

Select this radio button to interact with Dazzler using either GestPoint or ClearTouch.

Timeout

If for some reason the Mouse Tracker loses connection with GestPoint or ClearTouch, this specifies the amount of time (in seconds) before reconnection is attempted.

Retry

Specifies how many times the Mouse Tracker will attempt to reconnect with GestPoint or ClearTouch, if disconnection occurs.

Mouse

Select this radio button to interact with Dazzler using a mouse.

Framerate

Enter the desired framerate (in frames per second).

Output

Diameter

Specifies the diameter, in pixels, of the outline data that will be sent to Dazzler upon mouse action.

If while using Dazzler you wish to see the outline created by the Dazzler Mouse Tracker, press **Ctrl-T**. Pressing **Ctrl-T** again turns off the outline.

7.6 Displaying the mouse pointer

By default, when Dazzler is running full screen, the mouse pointer is not displayed. When using the Mouse Tracker, you can get an idea of where the mouse is positioned by pressing **Ctrl-T** to enable the tracker outline display within Dazzler. You will see a circle on the screen of the size specified in your Mouse Tracker setup. However, if you would prefer to avoid having to hit **Ctrl-T** every time you run Dazzler, or if you have some other reason for wanting to see your mouse pointer, the following steps should be taken.

To display the mouse pointer:

1. Open the **dazzler.cfg** file in a text editor such as Notepad (the **dazzler.cfg** file is located in the **Config** folder).

2. Find the following line:

HideMouseInFullScreen="True"

3. Change "True" to "False" and save the file.

To hide the mouse pointer:

1. Open the **dazzler.cfg** file in a text editor such as Notepad (the **dazzler.cfg** file is located in the **Config** folder).

2. Find the following line:

HideMouseInFullScreen="False"

3. Change "False" to "True" and save the file.

Care should be taken when editing the **dazzler.cfg** file as incorrect entries could render Dazzler inoperable. Create a copy of your existing file before editing.

Any changes to the **dazzler.cfg** file will affect Dazzler no matter which tracker is used.

8 Dazzler ActiveX Control

The **Dazzler ActiveX Control** allows you to display Dazzler within Internet Explorer.

The **Dazzler ActiveX Control** only works with Internet Explorer, and may require version 7 or higher.

8.1 Using Dazzler ActiveX Control

Insert the following code within the body section of the HTML page. Substitute where indicated.

```
<object
  id="idname " where idname is a unique identifier.
  width="Enter desired width in pixels or percentage"
  height="Enter desired height in pixels or percentage"
  classid="clsid:8905177F-FD9E-44CD-9DCF-15609C6529A8"
>
</object>

<script language="JavaScript" type="Text/Javascript">
  idname.SparkFile = "[Enter spark file name]"; where idname is the unique identifier specified a
</script>
```

The following example defines a **Dazzler ActiveX control** named, "ActiveX_sample". Width is 640 pixels. Height is 480 pixels. The **Dazzler ActiveX control** will play a Spark file named "mySpark. spark" that is located in a subfolder named "myPlayItems", which is located in the Resources folder.

```
<object
  id="ActiveX_sample"
  width="640"
  height="480"
  classid="clsid:8905177F-FD9E-44CD-9DCF-15609C6529A8"
>
</object>

<script language="JavaScript" type="Text/Javascript">
  ActiveX_sample.SparkFile = "myPlayItems\\mySpark.spark";
</script>
```

Notes

The Tracker needs to be running before running the **Dazzler ActiveX Control.**

When running the HTML page with the **Dazzler ActiveX Control** for the first time, you may get the following **ActiveX Warning** at the top of your Internet Explorer window:

> "To help protect your security, Internet Explorer has restricted this webpage from running scripts of ActiveX controls that could access your computer. Click here for options..."

If you get this message, click anywhere on that message. A small menu will appear. Select "**Allow blocked content**...".

A Security Warning may appear. Click on **Yes**.

8.2 Sample: ActiveX Control

Allows you to play Dazzler within Internet Explorer

This sample inserts the **Dazzler ActiveX Control** inside an HTML frame.

To view the HTML frame set as illustrated above, run the **index.htm** file located in your **Resources\Manual_Samples\ActiveX** folder. The code for the Dazzler ActiveX Control is in the **Dazzler1.htm** file in the this same folder.

The Tracker needs to be running before running the **Dazzler ActiveX Control.**

9 Customizing Effects - A Tutorial

This tutorial was designed for those who might like to customize the Dazzler effects that are provided with their GestureTek product. Anyone with a working knowledge of computers can learn to swap background images and change the look and feel of the existing effects. In this tutorial, you'll change a background image, customize an existing effect and then combine two existing effects and customize them to create a new effect. You'll also learn how to create a thumbnail image of your new effect to include in the GestureTek Control Panel. There are four tasks involved in this exercise:

- changing the background image
- changing the behavior of an existing effect
- combining and customizing two effects
- creating a thumbnail image of the new effect

In order to complete this tutorial, you will need to have a 1024 x 768 image, in one of the following image file formats:

- BMP
- DDS
- DIB
- HDR
- JPG
- PFM
- PNG
- PPM
- TGA

9.1 Getting started

To make it easier to view any changes you make as you're customizing an effect, first create a shortcut to Dazzler on the desktop image, following the instructions in this section. The Dazzler Demos and Manual_Samples themselves should never be changed. To swap a background image or customize an effect, it is best to copy the desired folders and files into a new folder. For this purpose, please set up folders for the new effects you will create, as described in this section.

9.1.1 To create a Dazzler shortcut on the desktop

1. Press **Esc** to exit Dazzler if it is running.

2. Click **Start**>**Programs**>**GestureTek**>**ProductName>**.

3. Right-click on **Dazzler** and select **Send To** from the menu.

4. Select **Desktop**.

▶ The Dazzler icon will now appear on your desktop image and as long as the tracker for your product is running, you will be able to click on it to view your effect.

9.1.2 To set up folders for new effects

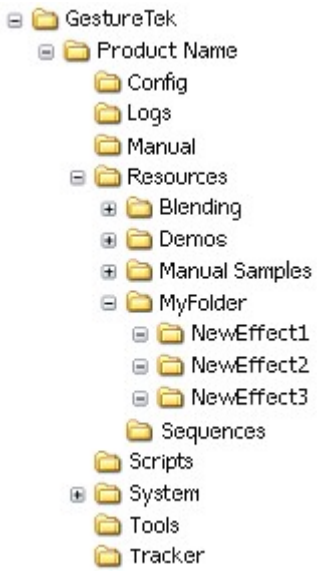
1. Navigate to the **Resources** directory (the default location is **C:\GestureTek\ProductName \Resources**) and create a new folder in the directory. Name it **MyFolder**.

Your new folder *must* be located in the **Resources** directory in order for Dazzler to be able to retrieve and run it.

2. In your new folder, create three more folders, **NewEffect1**, **NewEffect2** and **NewEffect3**.

File and folder names must only be comprised of letters, numbers, underscores and dashes.

▶ Your GestureTek directory structure should now look like this:


GestureTek
Product Name
Config
Logs
Manual
Resources
Blending
Demos
Manual Samples
MyFolder
NewEffect1
NewEffect2
NewEffect3
Sequences
Scripts
System
Tools
Tracker

9.2 Lesson 1: Changing the background image

The ability to change the background image is useful for creating an effect that has your own unique branding. This lesson guides you through the steps involved and includes instructions for playing your new effect.

9.2.1 To set up a new effect

1. Press **Esc** to exit Dazzler if it is running.

   If exiting Dazzler brings you to the Control Panel, click on **Exit** to get to the desktop.

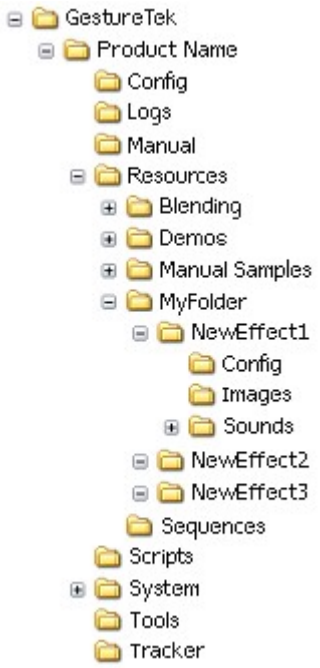
2. From the **Demos** or **Manual_Samples** folder in the **Resources** directory, browse to the effect whose background you'd like to change and open the folder.

   The default location of the **Resources** directory is **C:\GestureTek\ProductName**.

3. Copy all the folders and the **root.spark** file into the **NewEffect1** folder you created at the beginning of this tutorial.

   Only make changes to files in your new folder.

   Your GestureTek directory structure should now look something like this, depending on whether or not the effect you chose has a **Sounds** folder:

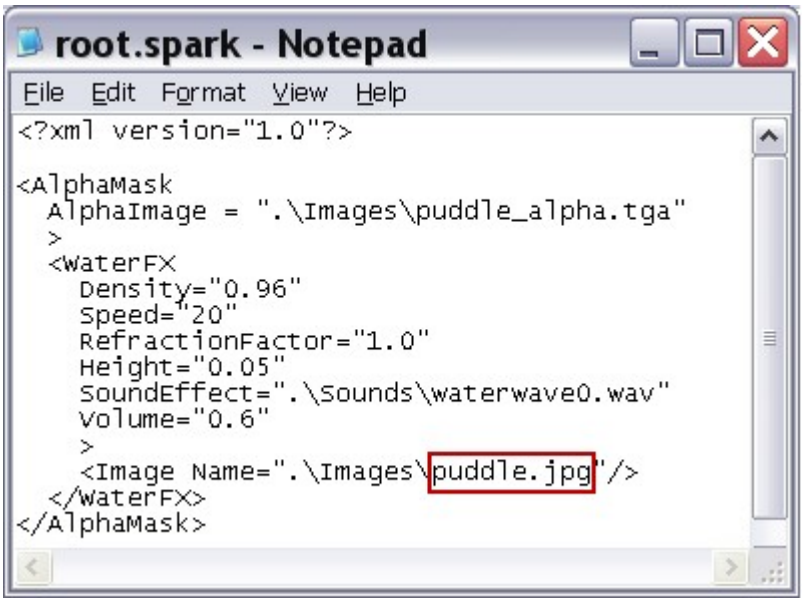


9.2.2 To change the background image

1. In your **NewEffect1** folder, add your new background image file to the **Images** folder.

   You can use any image you like, as long as it has a resolution of 1072 x 768 and is in one of the file formats listed in the introduction.

2. Open the **root.spark** file with Notepad or another text editor and replace the image file name with the file name of your new image. Image file names must be comprised of letters, numbers, underscores and dashes only. See image below.



3. Click **File>Save** and continue with To play the new effect 184.

9.2.3 To play the new effect

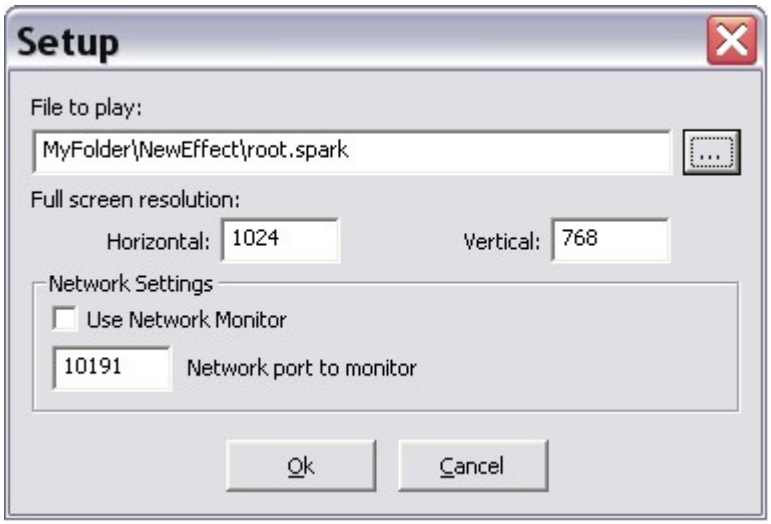
1. Double-click the Dazzler icon on your desktop to start Dazzler.

   If this is the first time you are playing your effect, continue with **Step 2**.

   If you have previously played your new effect, Dazzler will replay it now. Continue with **Step 5** below.

2. Right-click anywhere on the display and select **Setup** from the context menu.

   The **Setup** dialog box appears.



3. Click the browse button to the right of the **File to play** field and then, in the **Choose a file** dialog box, click on **MyFolder>NewEffect1>root.spark** and click **OK**.

   The path to the **root.spark** file of your new effect is now entered in the **File to play** field.

4. Click **OK**.

   Dazzler will now play your new effect.

5. If you are finished with your changes, close Dazzler and continue with the next lesson if desired or go to Lesson 4: Creating a thumbnail image of your new effect 192.

   Press **Esc** to close Dazzler and if necessary, click **Exit** on the Control Panel to get back to the desktop.

9.3 Lesson 2: Changing the behavior of an existing effect

Each effect has numerous attributes that can be changed to alter its behavior. Your new effect must follow the same format as the existing effects. The easiest way to make sure that the effect will work, is to base it on an existing effect. In this lesson, we'll change the **ParticlesEmitMotion1** effect to make a new effect.

9.3.1 To set up a new effect

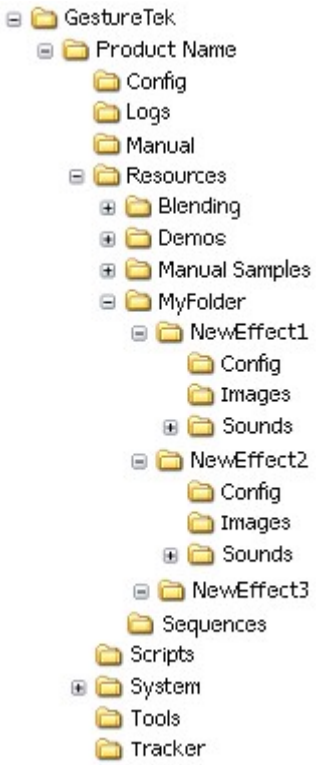
1. From the **Manual_Samples** folder in the **Resources** directory, browse to the **ParticlesEmitMotion1** sample and open the folder.

   The default location of the **Resources** directory is **C:\GestureTek\ProductName**.

2. Copy all the folders and the **root.spark** file into the **NewEffect2** folder you created at the beginning of this tutorial.

   Only make changes to files in your new folder.

   Your GestureTek directory structure should now look like this:



9.3.2 To change the behavior of an effect

1. Open the **root.spark** file in your **NewEffect2** folder with Notepad or another text editor and begin changing the attributes.

2. Decrease the **EmitRatePerSecond** attribute from "100" to "80" to generate fewer bubbles.

   EmitRatePerSecond was previously called **EmitRate** and is so called in the **ParticlesEmitMotion1** sample.

   You can play the effect after each change to see what impact your change has had, or make several changes before checking it. Click **File**>**Save** in order to save your changes before playing the effect. See To play the new effect. 184

3. Decrease the **MinWidth** attribute to "5" and the **MaxWidth** attribute to "30" to generate smaller bubbles that vary from 5 to 30 pixels in width.

4. Change the **FadeTime** to "1.5" to increase the length of time it takes for the bubbles to fade out.

5. Continue changing the values of the various attributes as desired, using the information in the table below.

   This table does not include all attributes for this effect; just those that would more commonly be changed. Information on all the attributes of all the effects can be found in the Play Items sections of this User Guide.

Attribute	Description
EmitRatePerSecond (or EmitRate)	Specifies the number of particles spawned per second, upon motion detection. Set the value higher to generate more particles, or lower to generate fewer. (EmitRate specifies the number of particles spawned *per frame*, but generates similar results.)
MinWidth/MaxWidth	The minimum and maximum width of each particle. When a particle is spawned, it appears, then grows to a random pixel size within these two limits. Setting both MinWidth and MaxWidth to the same value makes all particles the same size.
MaintainAspectRatio	When set to "True", the particle will always be the same height as the width generated by the MinWidth and MaxWidth values. Therefore, setting this value to "True" will make the bubbles perfectly round.
MinLiveTime/MaxLiveTime	The minimum and maximum time, in seconds, that a particle remains on-screen before beginning to fade out. The effect chooses a random time within the range. MaxLiveTime must be equal to or greater than MinLiveTime. Identical values will cause all particles to appear on-screen for the same amount of time before beginning to fade out. For a random effect, make the MaxLiveTime larger than the MinLiveTime.

Attribute	Description
FadeTime	The time, in seconds, it takes for a particle to fade out, once its LiveTime has expired. The smaller the value, the faster the particle fades out. A value of 0.0 will make the particle disappear instantly as soon as its live time has expired. Try setting a higher value, so that the bubbles fade more slowly.
Velocity	Sets the direction and speed that the particles will move upon spawning. The first value controls horizontal direction (a positive value denotes movement to the right; a negative value denotes movement to the left). The second value controls vertical direction (a positive value denotes downward movement; a negative value denotes upward movement). A setting of "0.0, 0.0" denotes no movement.
Acceleration	Sets the rate of increase of movement, in pixels per second, applied to the particles after spawning. The first value controls horizontal direction (a positive value denotes movement to the right; a negative value denotes movement to the left). The second value controls vertical movement (a positive value denotes downward movement; a negative value denotes upward movement). A setting of "0.0, 0.0" denotes no acceleration.Setting a positive value for horizontal movement in the Velocity attribute and a negative value for vertical acceleration, gives you bubbles that move to the right and upwards.
ExpandRate	Sets the time in seconds it takes for a particle to expand from the size at which it first appears to its full size (as defined in the MinWidth and MaxWidth attribute). The lower the value, the faster the particle expands. Setting the ExpandRate to "0.0" makes the particle appear at its full size with no expansion effect.

6. When you have finished making your changes and are happy with your new effect, continue with Lesson 3: Combining two effects 188 if desired or Lesson 4: Creating a thumbnail image of your new effect 192.

9.4 Lesson 3: Combining two effects

In this lesson, we'll apply a water effect over the **ParticlesEmitMotion1** effect used in the previous lesson, change the background image and then customize the effect.

9.4.1 To set up a new effect

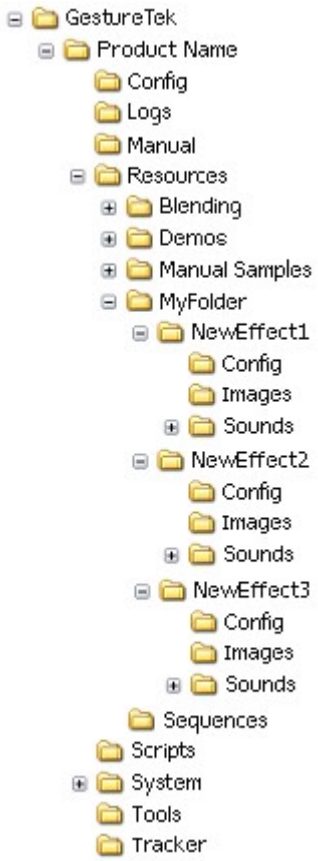
1. From the **Manual_Samples** folder in the **Resources** directory, browse to the **ParticlesEmitMotion1** sample and open the folder.

   The default location of the **Resources** directory is **C:\GestureTek\ProductName**.

2. Copy all the folders and the **root.spark** file into the **NewEffect3** folder you created at the beginning of this tutorial.

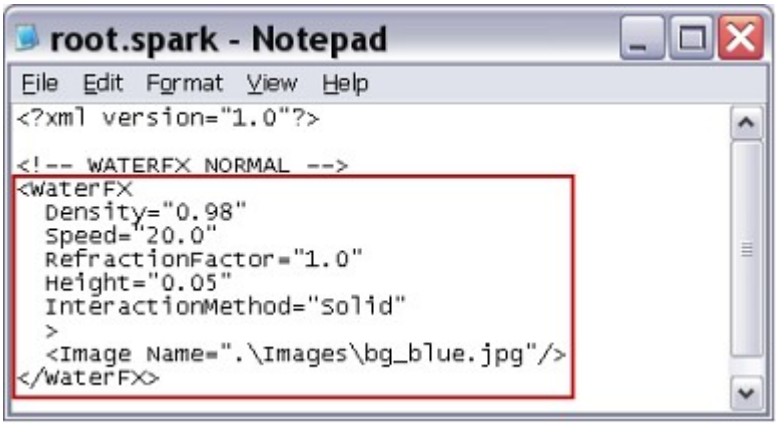
   Only make changes to files in your new folder.

   Your GestureTek directory structure should now look like this:



3. Now, from the **Manual_Samples** folder, open the **WaterFX_Sample1** folder and open the **root.spark** file with Notepad or another text editor.

4. Copy all the code in the **root.spark** file from the opening tag (**<WaterFX**) through to the closing tag (**</WaterFX>**).



5. In the **NewEffect3** folder, open the **root.spark** file and paste the **WaterFX** code into it, just before the closing tag of the **ParticlesEmitMotion** code.

Paste the code over the existing image file code line, thus over-writing the code. See image below.

6. Click **File>Save** to save your changes.

This is how the combined effect looks. Check your work by playing the effect. 184

9.4.2 To customize the new effect

1. Follow steps in To change the background image 183 in Lesson 1 to change the background image of the new effect using any 1024 x 768 image in one of the supported image file formats.

2. Open the **root.spark** file in the **NewEffect3** folder with Notepad or another text editor.

3. Change the values of the various attributes as in Lesson 2, referring to that table and/or the Play Items sections of this User Guide.

   You can play the effect after each change to see what impact your change has had, or make several changes before checking it. Click **File**>**Save** in order to save your changes before playing the effect. See To play the new effect 184.

4. When you are satisfied with the look of your new effect, continue with Lesson 4: Creating a thumbnail image of your new effect 192.

9.5 Lesson 4: Creating a thumbnail image of your new effect

1. Play your new effect on your computer.
2. Take a screenshot of the effect while it is running.
3. Save the screenshot to your desktop as **Config.jpg**.
4. Replace the image file in the **Config** folder of the **NewEffect1, 2 or 3** (as applicable) folder with the **Config.jpg** file you just created.
5. To view the thumbnail image, stop Dazzler if it is running, then click on the Control Panel icon on your desktop.
6. In the Control Panel, click on **MyFolder** in the **AVAILABLE EFFECTS** window and click on **NewEffect1** (or **NewEffect2** or **NewEffect3**).

▶ Now you will see a thumbnail image that is representative of the effect, in the image window. The image below is a combination of **ParticlesEmitMotion1** and **WaterFX_Sample1** with a new background image and some customization.

10 Troubleshooting

Problem	Solution
Dazzler ActiveX doesn't work.	Run **RegisterDazzlerX.bat** (from the **System** folder).
Display screen is black.	You may have entered an unsupported resolution. 1. Right-click on the screen and select **Setup**. 2. Enter the correct resolution (default is Horizontal: 1024 and Vertical: 768) and click **OK**. Alternatively, go to **C:\Gesturetek\GestureTekProduct\Config** and open the **Dazzler.cfg** file with Notepad. Edit the **FullScreenHeight** and **FullScreenWidth** values as necessary. Save your changes.

11 GestureTek Contact Information

GestureTek Website: http://www.gesturetek.com

GestureTek Addresses: http://www.gesturetek.com/aboutus/worldwide-locations.php

Technical Support: http://www.gesturetek.com/support/index.php

Technical Support (Tel.): 1-800-315-1189 (Toll Free) or 1-416-340-9290, ext. 225